Universität Hildesheim
Fachbereich III
Informations- und Kommunikationswissenschaften

Arbeit zur Erlangung des akademischen Grades einer
Magistra Artium
im Fach Internationales Informationsmanagement
Schwerpunkt Informationswissenschaften

Entwicklung eines Werkzeugs zur Sprachidentifikation in mono- und multilingualen Texten.

Hildesheim, im August 2005

Erstgutachterin:
Prof. Dr. Christa Womser-Hacker

Zweitgutachter:
Dr. Thomas Mandl

Vorgelegt von: Olga Artemenko
olga-art@gmx.de

Margaryta Shramko
m.shramko@gmx.de

Abstract

Identifikation der Sprache bzw. Sprachen elektronischer Textdokumente ist einer der wichtigsten Schritte in vielen Prozessen maschineller Textverarbeitung. Die vorliegende Arbeit stellt LangIdent, ein System zur Sprachidentifikation von mono- und multilingualen elektronischen Textdokumenten vor. Das System bietet sowohl eine Auswahl von gängigen Algorithmen für die Sprachidentifikation monolingualer Textdokumente als auch einen neuen Algorithmus für die Sprachidentifikation multilingualer Textdokumente.

Identification of the language or languages of a document is one of the most important steps in the automatic text processing. This work presents a novel system, called LangIdent, which provides the language identification for mono- and multilingual documents. The system includes both well known algorithms for the language identification for monolingual documents and a new developed algorithm identifying the languages of multilingual documents.

Inhaltsverzeichnis

Einführung

Mit der Verbreitung des Internets vermehrt sich die Menge der im World Wide Web verfügbaren Dokumente. Die Gewährleistung eines effizienten Zugangs zu gewünschten Informationen für die Internetbenutzer wird zu einer großen Herausforderung an die moderne Informationsgesellschaft. Eine Vielzahl von Werkzeugen wird bereits eingesetzt, um den Nutzern die Orientierung in der wachsenden Informationsflut zu erleichtern. Allerdings stellt die enorme Menge an unstrukturierten und verteilten Informationen nicht die einzige Schwierigkeit dar, die bei der Entwicklung von Werkzeugen dieser Art zu bewältigen ist. Die zunehmende Vielsprachigkeit von Web-Inhalten resultiert in dem Bedarf an Sprachidentifikations-Software, die Sprache/en von elektronischen Dokumenten zwecks gezielter Weiterverarbeitung identifiziert. Solche Sprachidentifizierer können beispielsweise effektiv im Bereich des Multilingualen Information Retrieval eingesetzt werden, da auf den Sprachidentifikationsergebnissen Prozesse der automatischen Indexbildung wie Stemming, Stoppwörterextraktion etc. aufbauen.

In der vorliegenden Arbeit wird das neue System „LangIdent“ zur Sprachidentifikation von elektronischen Textdokumenten vorgestellt, das in erster Linie für Lehre und Forschung an der Universität Hildesheim verwendet werden soll. „LangIdent“ enthält eine Auswahl von gängigen Algorithmen zu der monolingualen Sprachidentifikation, die durch den Benutzer interaktiv ausgewählt und eingestellt werden können. Zusätzlich wurde im System ein neuer Algorithmus implementiert, der die Identifikation von Sprachen, in denen ein multilinguales Dokument verfasst ist, ermöglicht. Die Identifikation beschränkt sich nicht nur auf eine Aufzählung von gefundenen Sprachen, vielmehr wird der Text in monolinguale Abschnitte aufgeteilt, jeweils mit der Angabe der identifizierten Sprache.

Die Arbeit wird in zwei Hauptteile gegliedert. Der erste Teil besteht aus Kapiteln 1-5, in denen theoretische Grundlagen zum Thema Sprachidentifikation dargelegt werden. Das erste Kapitel beschreibt den Sprachidentifikationsprozess und definiert grundlegende Begriffe. Im zweiten und dritten Kapitel werden

vorherrschende Ansätze zur Sprachidentifikation von monolingualen Dokumenten dargestellt und miteinander verglichen, indem deren Vor- und Nachteile diskutiert werden. Das vierte Kapitel stellt einige Arbeiten vor, die sich mit der Sprachidentifikation von multilingualen Texten befasst haben. Der erste Teil der Arbeit wird mit einem Überblick über die bereits entwickelten und im Internet verfügbaren Sprachidentifikationswerkzeuge abgeschlossen.

Der zweite Teil der Arbeit stellt die Entwicklung des Sprachidentifikationssystems LangIdent dar. In den Kapiteln 6 und 7 werden die an das System gestellten Anforderungen zusammengefasst und die wichtigsten Phasen des Projekts definiert. In den weiterführenden Kapiteln 8 und 9 werden die Systemarchitektur und eine detaillierte Beschreibung ihrer Kernkomponenten gegeben. Das Kapitel 10 liefert ein statisches UML-Klassendiagramm mit einer ausführlichen Erklärung von Attributen und Methoden der im Diagramm vorgestellten Klassen. Das nächste Kapitel befasst sich mit den im Prozess der Systementwicklung aufgetretenen Problemen. Die Bedienung des Programms wird im Kapitel 12 beschrieben. Im letzten Kapitel der Arbeit wird die Systemevaluierung vorgestellt, in der der Aufbau und Umfang von Trainingskorpora sowie die wichtigsten Ergebnisse mit der anschließenden Diskussion präsentiert werden.

I Theoretische Grundlagen und Literaturüberblick

Im ersten Teil der Arbeit werden theoretische Grundlagen vermittelt. Es werden aus der wissenschaftlichen Literatur bekannte und meist verbreitete Ansätze zur Sprachidentifikation monolingualer Dokumente vorgestellt und miteinander verglichen, sowie einige Verfahren zur Sprachidentifikation von multilingualen Texten beschrieben. Anschließend wird ein Überblick über im Internet verfügbare Werkzeuge zur automatischen Sprachidentifikation gegeben.

1 Grundlegendes zu Sprachidentifikation

Folgendes Kapitel liefert eine Einführung in das Thema der Sprachidentifikation. Es definiert grundlegende Begriffe, die für die theoretische Darlegung des Themas unumgänglich sind, gibt eine schematische Darstellung eines Sprachidentifikationsprozesses mit einer kurzen Erläuterung dessen wichtigsten Phasen und schafft einen Überblick über die wissenschaftlichen Beiträge auf diesem Gebiet.

1.1 Einführung und grundlegende Begriffe

Unter dem Begriff **automatische Sprachidentifikation** wird in der vorliegenden Arbeit ein Prozess verstanden, in dem die Sprache/en eines elektronischen Textdokuments mittels einer dafür speziell entwickelten Software ermittelt wird/werden. Es handelt sich in diesem Fall um die Identifikation *geschriebener* und nicht gesprochener Sprache. Das Letztere wird in der Fachliteratur als automatische Spracherkennung (engl. Speech Recognition) bezeichnet.
Im Allgemeinen kann der Prozess der automatischen Sprachidentifikation in zwei Hauptphasen eingeteilt werden: die Trainingsphase (engl. modelling stage) und die Erkennungsphase (engl. the classification stage) (vgl. POUTSMA, A. 2001: 2). Schematisch kann der Prozess wie folgt dargestellt werden:

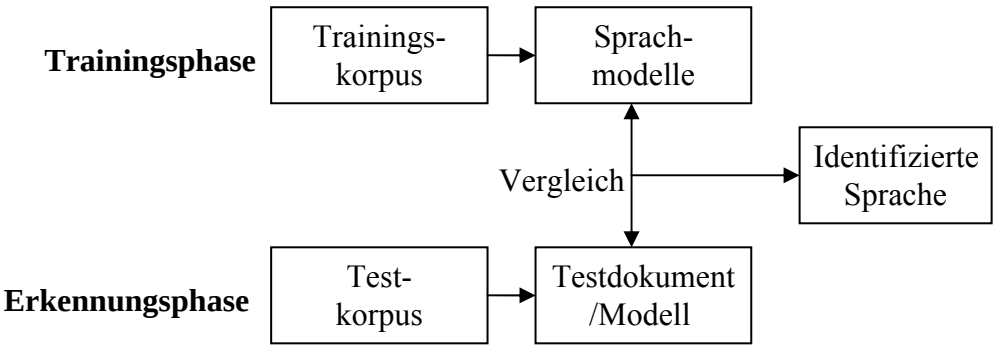


Abbildung 1: Phasen des Sprachidentifikationsprozesses

In der ersten Phase findet ein **Systemtraining** statt. Darunter wird ein überwachter Lernprozess verstanden, in dem das System die typischen Merkmale einer Sprache lernt.

Für diese wichtige Phase wird ein **Trainingskorpus** benötigt, der eine Sammlung von in elektronischer Form vorliegenden Dokumenten darstellt, die bezüglich deren Sprache bereits vorklassifiziert sind. Die Trainingsdokumente müssen dabei alle Sprachen abdecken, die später durch das System identifiziert werden sollen.

Die typischen Merkmale einer Sprache werden anhand der im Korpus enthaltenen Dokumente erlernt und in einem so genannten **Sprachmodell** zusammengefasst.

In der Erkennungsphase werden Sprachen aller Dokumente aus dem Testkorpus bestimmt. Ein **Testkorpus** ist eine Sammlung von Textdokumenten, die bezüglich ihrer Sprache vom System klassifiziert werden müssen. In den meisten Ansätzen werden dafür so genannte Dokumentmodelle erzeugt, die im nächsten Schritt mit den vorhandenen Sprachmodellen verglichen werden. Mittels eines **Klassifikationsverfahrens** wird das Testdokument bzw. Testdokumentmodell demjenigen Sprachmodell zugeordnet, das am wahrscheinlichsten die Dokumentsprache repräsentiert.

Der genaue Verlauf des Sprachidentifikationsprozesses und eine detaillierte Beschreibung der darin angewandten Methoden, wie Sprachmodellierungs- und Klassifikationsmethoden, werden in den nachfolgenden Kapiteln dargestellt. Für das Verständnis dieser Kapitel, vor allem für die Beschreibung von Sprachmodellierungsmethoden, müssen einige weitere Begriffe wie absolute

Häufigkeit, relative Häufigkeit, kumulative Häufigkeit und inverse Dokumenthäufigkeit definiert werden. Die ersten drei Termini stammen aus dem Bereich der deskriptiven Statistik und werden in vielen Ansätzen der Sprachmodellierung eingesetzt.

Absolute Häufigkeit ist ein Maß, das angibt, wie viele Merkmalsträger zu einer bestimmten Merkmalsausprägung in einem Datensatz existieren (vgl. http://de.wikipedia.org/ /wiki/Absolute_H%C3%A4ufigkeit, Zugriff 12.05.2005, 13:10 MEZ). Im Kontext der vorliegenden Arbeit wird darunter die Anzahl des Auftretens $H_d(x)$ einer Spracheinheit x in einem Trainings- bzw. Testdokument d verstanden.

Die **relative Häufigkeit** wird berechnet, um Häufigkeitsverteilungen bezüglich der Dimension unterschiedlicher Grundgesamtheiten besser abwägen zu können (vgl. http://de. wikipedia.org/wiki/Relative_H%C3%A4ufigkeit, Zugriff 12.05.2005, 13:10 MEZ).

Die relative Häufigkeit einer Spracheinheit in einem Trainings- bzw. Testdokument wird berechnet, indem die absolute Häufigkeit $H_d(x)$ der Spracheinheit x im Trainings- bzw. Testdokument d durch die Anzahl N_d aller im Trainings- bzw. Testdokument d vorkommenden Spracheinheiten dividiert wird:

$$RH_d(x) = \frac{H_d(x)}{N_d}$$

Kumulative Häufigkeit wird ermittelt, indem man sukzessiv die absoluten bzw. relativen Häufigkeiten aufeinander folgender Messwerte (*hier*: Häufigkeiten aufeinander folgender Spracheinheiten in einem Sprachmodell) addiert (vgl. http://vs.fernuni-hagen.de/Methoden/ILS/LS/ Glossar/kumulative_Haeufigkeit. html, Zugriff 26.05.2005, 18:10 MEZ).

Der Begriff **inverse Dokumenthäufigkeit** (*idf*) wird im Bereich von Information Retrieval verwendet und „dient […] zur Bestimmung der Trennfähigkeit eines Wortes bzw. Termes für die Indexierung von Dokumenten. Ein Wort, das nur in wenigen Dokumenten oft vorkommt ist geeigneter als eines, das in fast jedem Dokument oder nur sehr gering auftritt“ (http://de.wikipedia.org/wiki/Inverse_

Dokumenth%C3%A4ufigkeit, Zugriff 12.05.2005, 13:06 MEZ). Die inverse Dokumenthäufigkeit wird wie folgt berechnet:

$$idf_d(x) = \frac{H_d(x)}{n}$$

wobei $H_d(x)$ die Häufigkeit der Spracheinheit x im Dokument d bezeichnet und n die Anzahl der Dokumente, in denen x vorkommt.

1.2 Stand der Forschung

Das Problem der Sprachidentifikation von Textdokumenten wurde bereits in den 60er Jahren angegangen. Die in den vergangenen Jahrzehnten veröffentlichten Arbeiten umfassen eine Vielfalt von analytischen Verfahren, die von den manuellen über semiautomatische zu den vollautomatischen Techniken reichen.
Einige der frühen Arbeiten (N. C. Ingle (1976), P. Newman (1987)) stellten eine Art Guides oder Handbüchern dar, die Bibliothekare und Übersetzer bei der Ermittlung der Sprache von Büchern bzw. gedruckten Dokumenten unterstützen sollten. Das Verfahren von N. C. Ingle beruht zum Beispiel auf der Beobachtung, dass die Präsenz bzw. die Abwesenheit von kurzen, aus einem oder zwei Buchstaben bestehenden Wörtern in vielen Fällen genügend ist, um 17 Sprachen des romanischen Alphabets voneinander unterscheiden zu können (vgl. SIBUN, P. & SPITZ, L. A. 1994: 19).
Ein kurzes Beispiel, wie mit Hilfe der von N. C. Ingle zusammengestellten Tabelle die Sprache eines Texts identifiziert werden kann (vgl. http://www.csee.umbc.edu/ ~dean3/cmsc691b/dw691b.pdf, S. 19. Zugriff 19.05.2005, 19:03 MEZ):
*„Naar aanleiding van **uw** brief van januari j.l. waarin **u** ons vraagt wat bedoelt wordt met het woord "kalf" **of** "achterkalf" kunnen wij **u** het volgende mededelen"*.
Anhand der Tabelleneinträge lassen sich im Text gefundene Kurzwörter folgenden Sprachen zuordnen:
uw – Niederländisch,

u – Niederländisch, Serbokroatisch, Polnisch, Tschechisch,
of – Niederländisch, Englisch.
Dementsprechend wird Niederländisch als Dokumentsprache ausgewählt.
Mit der fortschreitenden Entwicklung der Computertechnik, der Verbreitung des Internets und der ständig wachsenden Menge an zugänglichen Informationen in elektronischer Form kommt der automatischen Identifikation von elektronischen Dokumenten immer größere Bedeutung zu. Deshalb bezieht sich die überwiegende Anzahl der wissenschaftlichen Beiträge auf die Entwicklung von computergestützten automatischen Systemen zur Sprachidentifikation von elektronischen Textdokumenten.
Mit dem Problem der automatischen Sprachidentifikation befassten sich u. a. folgende Forscher: W. B. Cavnar & J. M. Trenkle (1994), C. Souter et al. (1994), T. Dunning (1994), P. Sibun & L. A. Spitz (1994), P. Sibun & J. C. Reynar (1996), G. Grefenstette (1995), M. Damashek (1995), M. J. Martino & R. C. Paulsen (1996, 1999, 2001), J. M. Prager (1999), J. Cowie et al. (1999), C. L. Tan (1999), B. M. Schulze (2000), R. D. Lins & P. Gonçalves (2004). In den Arbeiten der oben genannten Wissenschaftler werden Sprachidentifikationssysteme präsentiert, die auf unterschiedlichen Sprachmodellierungs- und Klassifikationsmethoden basieren, eine variierende Anzahl der Sprachen unterstützen und in ihrer großen Mehrheit gute Erkennungsquoten erreichen. Auf die Methoden und erreichte Genauigkeit der Sprachidentifikationssysteme wird in den nachfolgenden Kapiteln der vorliegenden Arbeit ausführlich eingegangen.

2 Methoden der Sprachidentifikation

In diesem Kapitel werden mehrere Ansätze zur automatischen Sprachidentifikation und deren Realisierungen ausführlich dargestellt, wobei der Schwerpunkt auf die Beschreibung von Sprachmodellierungsmethoden gelegt wird. Einer detaillierten Darlegung der Klassifikationsverfahren wird das nachfolgende Kapitel gewidmet.

In der gegenwärtigen wissenschaftlichen Forschung lassen sich zwei vorherrschende Ansätze zur Sprachidentifikation voneinander abgrenzen (vgl. LANGER, S. 2002: 1):

- N-Gram basierter Ansatz
- Wortbasierter Ansatz

Die beiden Ansätze unterscheiden sich voneinander hauptsächlich dadurch, dass sie verschiedene Spracheinheiten (engl. *features*) anwenden, anhand deren Frequenzen Modelle für die zu identifizierenden Sprachen erzeugt werden. Beim N-Gramm basierten Ansatz erfolgt die Sprachidentifikation auf der Basis der Häufigkeit von N-Grammen unterschiedlicher Länge. Beim wortbasierten Ansatz wird die Sprache des Dokuments anhand von Wörterbüchern (Lexika) ermittelt, die üblicherweise hoch bis mittel frequente Wörter der jeweiligen Sprache enthalten.

In der Forschung sind auch andere Verfahren zur Sprachidentifikation bekannt. Zu nennen sind, zum Beispiel, Sprachidentifikation auf der Basis von für die jeweilige Sprache typischen Sonderzeichen (z. B. ä, ü, ĉ, ç, ß, ï etc.) oder Sprachidentifikation mittels der Auftretenswahrscheinlichkeit der unikalen Buchstabenkombinationen (z.B. „sch“ im Deutschen, „czy“ im Polnischen etc.). Diese Verfahren werden in der vorliegenden Arbeit nicht näher behandelt, da sie sich im Vergleich zu den oben vorgestellten Ansätzen als eher uneffektiv erwiesen haben (vgl. SOUTER, C. et al. 1994: 183) und deswegen wenig angewendet worden sind.

Ein weiteres Verfahren basiert auf der Analyse von so genannten „word shape tokens“ (SIBUN, P. & SPITZ, L. A. 1994; SIBUN, P. & REYNAR, J.C. 1996; TAN, C. L. 1999). Der Ansatz beruht auf der Beobachtung, dass Wörter einer Sprache typische Umrissmuster haben, die in dieser Sprache häufiger als in anderen

vorkommen. Diese Methode wurde ursprünglich für die Sprachidentifikation von gedruckten Dokumenten und später auch für die Sprachidentifikation von Dokumenten in elektronischer Form eingesetzt. Wegen der besonderen Herangehensweise an das Problem der Sprachidentifikation wird die Methode im Abschnitt 2.8 kurz vorgestellt.

2.1 N-Gramm basierter Ansatz

Die Hauptidee des N-Gramm basierten Ansatzes besteht in der Verwendung von kleinen Spracheinheiten, so genannten N-Grammen, für die Sprachmodellbildung und ferner für die Sprachidentifikation von Dokumenten.
Unter einem **N-Gramm** wird eine Sequenz von *n* aufeinander folgenden Zeichen eines längeren Strings bzw. Wortes verstanden (vgl. CAVNAR, W. B. & TRENKLE, J. M. 1994: 2). Die Länge des kürzesten N-Gramms ist *n* = 1 und ist einem einzelnen Zeichen bzw. einem Buchstaben gleichzusetzen. In den wissenschaftlichen Experimenten werden normalerweise N-Gramme bis zu einer Maximallänge von fünf Zeichen verwendet (vgl. CAVNAR, W. B. & TRENKLE, J. M. 1994, SOUTER, C. et al. 1994, GREFENSTETTE, G. 1995, PRAGER, J. M. 1999 etc.). Das Wort „Information“ kann beispielsweise in folgende N-Gramme der Länge eins bis fünf aufgeteilt werden:

Uni-Gramme: i, n, f, o, r, m, a, t, i, o, n
Bi-Gramme: in, nf, fo, or, rm, ma, at, ti, io, on
Tri-Gramme: inf, nfo, for, orm, rma, mat, ati, tio, ion
Quad-Gramme: info, nfor, form, orma, rmat, mati, atio, tion
Quint-Gramme: infor, nform, forma, ormat, rmati, matio, ation

Im ersten Schritt der Trainingsphase wird jedes Trainingsdokument in N-Gramme zerlegt, so wie es oben dargestellt ist. In einigen Arbeiten wird jeder Wortanfang und jedes Wortende mit einem Unterstrich („_“) versehen, um die exakten Wortgrenzen bestimmen zu können. In diesem Fall werden Unterstriche zu Bestandteilen von N-Grammen (vgl. CAVNAR, W. B. & TRENKLE, J. M. 1994: 2; SOUTER, C. et al. 1994: 187; GREFENSTETTE, G. 1995: 2).

Im nächsten Schritt kommt es zur eigentlichen Modellierung aller im Trainingskorpus enthaltenen Sprachen. In vielen Publikationen wird ein N-Gramm basiertes Sprachmodell als eine Liste von Tupeln beschrieben, wobei jedes Tupel ein N-Gramm und dazugehörige Charakteristika enthält. Für die Berechnung von Charakterisitika sind mehrere Ansätze bekannt, die im Folgenden kurz erläutert werden:

- Ermittlung der *relativen Häufigkeit* der N-Gramme in einer Sprache, indem die absolute Häufigkeit des jeweiligen N-Gramms durch die Anzahl aller in dem Trainingsdokument enthaltenen N-Gramme geteilt wird (DAMASHEK, M. 1995: 843; GREFENSTETTE, G. 1995: 2).
- Erstellung von einer *N-Gramm Rangliste*: N-Gramme werden in der umgekehrten Reihenfolge nach ihrer absoluten Häufigkeit sortiert und den Rangziffern zugeordnet. In diesem Fall bekommt das häufigste N-Gramm den Rang 1, das zweithäufigste den Rang 2 usw. (vgl. CAVNAR, W. B. & TRENKLE, J. M. 1994: 3ff).
- Absolute Häufigkeiten von N-Grammen aus dem Trainingskorpus einer Sprache werden mit deren *inversen Dokumenthäufigkeiten idf* multipliziert, wobei $idf = 1 / n_{i,}$ und n_i die Anzahl aller Sprachmodelle aus dem Trainingskorpus darstellt, in denen das jeweilige N-Gramm vorkommt (vgl. PRAGER, J. M. 1999: 3).

Eine alternative Vorgehensweise für die Sprachmodellierung stellt die Anwendung von Markov-Ketten dar (DUNNING, T. 1994). Die theoretische Grundlage dieser Methode und ihre Funktionsweise werden im Abschnitt 3.3.2 vorgestellt.

Nachdem das System die Charakteristika für N-Gramme in jeder Sprache gelernt hat, werden in der Erkennungsphase die eingegebenen Testdokumente oder ihre Modelle mit den erzeugten Sprachmodellen verglichen.

Die in der Praxis anwendbaren Methoden zum Vergleich von Sprach- und Dokumentmodellen werden in einer detaillierten Weise im Kapitel 3 erläutert.

2.2 Überblick über Realisierungen des N-Gramm basierten Ansatzes

Im Folgenden werden einige Arbeiten vorgestellt, die sich mit der auf dem N-Gramm basierten Ansatz zur Sprachidentifizierung befasst haben: W. B. Cavnar & J. M. Trenkle 1994, T. Dunning 1994, M. Damashek 1995 und J. M. Prager 1999. Der Beschreibung der Methode von W. B. Cavnar & J. M. Trenkle wird ein getrenntes Unterkapitel gewidmet, da sie zum einen in vielen Arbeiten referenziert und zum anderen bei der Implementierung von LangIdent verwendet wird.

2.2.1 Der Algorithmus von W. B. Cavnar & J. M. Trenkle

In der Arbeit von W. B. Cavnar & J. M. Trenkle wird ein auf der N-Gramm Methode basiertes System repräsentiert, das ursprünglich für die Textkategorisierung entwickelt und später auch bei der Sprachidentifikation eingesetzt und getestet wurde. Die Funktionsweise des Systems veranschaulicht das nachfolgende Schema:

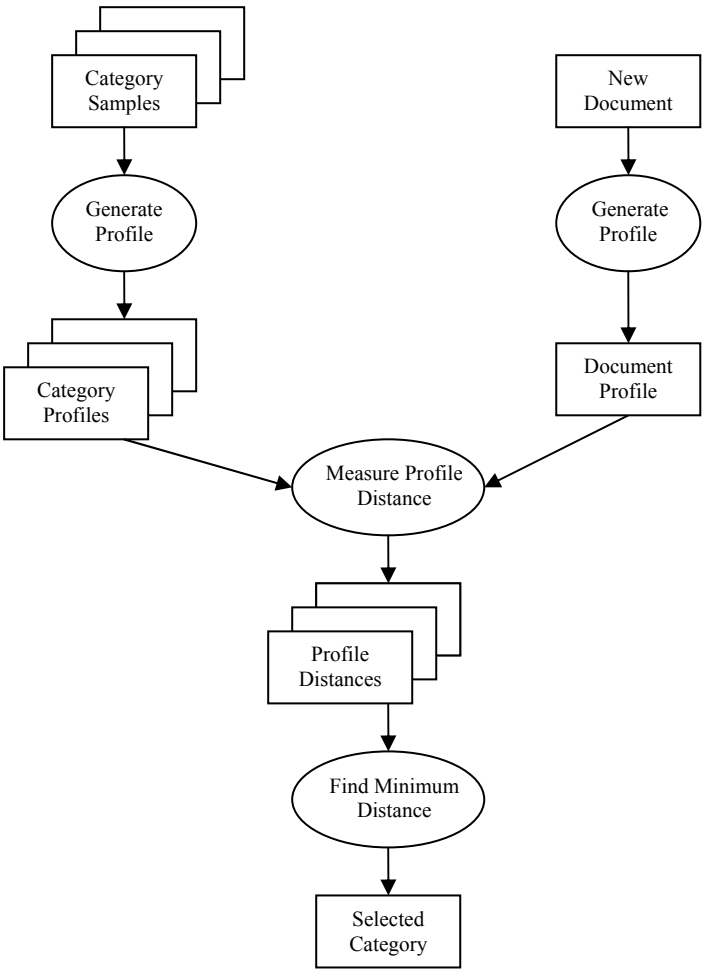


Abbildung 2: Datenfluss bei der N-Gramm basierten Textkategorisierung (CAVNAR, W. B. & TRENKLE, J. M 1994: 4)

Die Generierung von Profilen bezieht sich sowohl auf Dokumente aus dem Trainingskorpus als auch auf zu identifizierende Texte und beinhaltet folgende Schritte (vgl. CAVNAR, W. B. & TRENKLE, J. M. 1994: 3ff):

1. Schritt: Tokenisierung des Textes, d.h. Aufteilung des Textes in einzelne Wörter (Tokens), die nur aus Buchstaben und Apostrophen bestehen, Eliminierung von Zahlen und Satzzeichen. Für eine exakte Bestimmung von Wortgrenzen werden Wortanfänge und -enden mit Unterstrichen markiert.

2. Schritt: Zerteilung der Tokens in mögliche N-Gramme der Länge $n = 1$ bis $n = 5$. Das Wort „Text“ wird zum Beispiel in folgende N-Gramme der Länge $n = 2$ bis $n = 4$ geteilt:

Bi-Gramme: _t, te, ex, xt, t_

Tri-Gramme: _te, tex, ext_, xt_, t_ _

Quad-Gramme: _tex, text, ext_, xt_ _, t_ _ _

In diesem Beispiel werden die letzten N-Gramme des Wortes mit Unterstrichen ergänzt, so dass Folgendes gilt: ein Wort der Länge *k* (hier *k* = 4) ist in *k+1* Bi-Gramme, Tri-Gramme und Quad-Gramme zu zerlegen (hier 4+1=5).

3. Schritt: Berechnung der absoluten Häufigkeiten der N-Gramme.

4. Schritt: Zusammenfassung aller N-Gramme in einer Rangfolge, in der die N-Gramme in der umgekehrten Reihenfolge nach ihrer absoluten Häufigkeit geordnet werden.

Das Ergebnis der ersten vier Schritte sind so genannte „*category profiles*“ für Sprachen bzw. Textkategorien (sie werden im Folgenden als Sprachprofile bezeichnet) und „*document profiles*“ für zu identifizierende Texte (im Weiteren Dokumentprofile genannt).

Sprachprofile werden auf die ersten 300 häufigsten N-Gramme reduziert, da laut den Beobachtungen der Autoren die hoch frequenten N-Gramme sprachspezifisch sind (vgl. CAVNAR, W. B. & TRENKLE, J. M. 1994: 4). In den ersten Reihen befinden sich zumeist Uni-Gramme, die die Verteilung von Buchstaben des jeweiligen Alphabets im Dokument darstellen. Auf Uni-Gramme folgen Funktionswörter und hoch frequente Präfixe und Suffixe. Weniger häufige N-Gramme (die nach den ersten 300 folgen) repräsentieren dokument- bzw. themenspezifische Begriffe und Stämme, und es kann angenommen werden, dass sie von geringerem Interesse für die Identifikation der Dokumentsprache sind.

Die erstellten Sprach- und Dokumentprofile werden im nächsten Schritt miteinander verglichen, indem die Ad Hoc Ranking Methode angewendet wird. Eine ausführliche Darstellung der Methode folgt im Kapitel 3 (S. 32), wo ein zusammenfassender Überblick über die vorherrschenden Vergleichsansätze gegeben wird.

Der von W. B. Cavnar & J. M. Trenkle entwickelte Algorithmus wurde auf einem Korpus von 3713 Dokumenten in 14 Sprachen getestet, die aus einer „soc.culture“ Newsgroup stammen. In der Testphase wurden Dokumente unterschiedlicher Länge verwendet. Die Autoren experimentieren auch mit Sprachprofilen unterschiedlicher Länge: der Umfang variierte von 100 bis zu 400 N-Grammen. Im Laufe der Experimente stellte sich heraus, dass das System

weniger sensitiv zur Länge der Testdokumente als zur Länge der Sprachprofile ist. Das heißt: bei den Testdokumenten der gleichen Länge verbesserte sich die Erkennungsquote mit Umfangsvergrößerung von Sprachprofilen. Die beste Erkennungsquote betrug 99,8% für die Dokumentlänge von mehr als 300 N-Grammen und den Sprachprofilumfang von 300 und 400 N-Grammen.
Die Arbeit von W. B. Cavnar & J. M. Trenkle ist in der Hinsicht bedeutend, dass sich die darin vorgestellte Methode ziemlich einfach implementieren lässt und wenig Trainingsaufwand erfordert. Außerdem ist das entwickelte Sprachidentifikationssystem tolerant gegenüber Schreibfehlern, die oft in E-Mails, Forenbeiträgen und in Ergebnissen der *Optical Character Recognition* Systeme vorkommen können (vgl. CAVNAR, W. B. & TRENKLE, J. M 1994: 1).
Allerdings gibt es auch einige Schwachstellen. Die oben erwähnte Erkennungsquote von 99,8% bezieht sich auf Texte der Länge von 300 und mehr N-Grammen; bei der Spracherkennung von kürzeren Dokumenten hat sich das System jedoch nicht als besonders effektiv erwiesen. Aufgrund der Tokenisierung der Dokumente im ersten Schritt der Modellerstellung lässt sich das System auf Texte in den Sprachen, wo die Wortgrenzen schwer zu identifizieren sind (wie z.B. im Japanischen), nicht anwenden.
Der hier dargestellte Algorithmus wird von einigen Forschern entweder als Grundlage zur Entwicklung eigener Systeme genommen (KIKUI, G. 1996b; CAPSTIK, J. et al. 1999; MARTINS, B. & SILVA, M. J. 2005) oder er wird implementiert, um seine Effektivität mit eigenen Algorithmen zu vergleichen (COWIE, J. et al. 1999).

2.2.2 Weitere Realisierungen des N-Gramm basierten Ansatzes

Der N-Gramm basierte Ansatz findet seine Anwendung in der Arbeit von J. M. Prager 1999, die den Sprachidentifizierer *Linguini* präsentiert. Die in der Arbeit dargestellten Experimente zeigen, wie sich die für den Aufbau der Sprachmodelle genutzten N-Gramme unterschiedlicher Länge: Bi-Gramme, Tri-Gramme, Quad-Gramme und Quint-Gramme auf die Erkennungsqualität des Sprachidentifizierers auswirken. Im Unterschied zur Methode von W. B. Cavnar

& J. M. Trenkle bestehen die erstellten Sprachmodelle aus N-Grammen einer festen Länge.

Ein weiterer Unterschied besteht darin, dass in Linguini für den Vergleich von Testdokumenten mit den erzeugten Sprachmodellen das Vektorraum Modell eingesetzt wird. Um die Sprache eines Testdokuments zu bestimmen, werden als erstes zu identifizierende Dokumente und Sprachmodelle als Vektoren dargestellt. Es wird angenommen, dass das Dokument in der Sprache verfasst ist, deren Modell-Vektor den kleinsten Winkel zum Testdokument-Vektor bildet. Die Methode wird im Unterkapitel 3.2 genauer beschrieben.

Die durchgeführten Tests lieferten folgende Ergebnisse:

	Chunk Size (Byte)					
Feature-set	20	50	100	200	500	1000
2-grams	68,8	86,2	93,5	97,7	98,8	100,0
3-grams	79,5	93,0	97,7	99,3	100,0	100,0
4-grams	83,6	94,3	98,2	99,6	99,9	100,0
5-grams	81,4	93,1	97,8	99,4	99,9	99,9

Tabelle 1: Linguini's Erkennungsquoten bei der Verwendung von N-Grammen unterschiedlicher Länge (vgl. PRAGER, J. M. 1999: 5)

Aus der oben dargestellten Tabelle lässt sich erkennen, dass die auf den Quad-Gramm-Frequenzen basierenden Modelle die besten Ergebnisse erzielt haben. Die allgemeine Tendenz zur Verbesserung der Erkennungsqualität wird bei der Vergrößerung des Testdokumentumfangs (*chunk size*) beobachtet. Eine fast optimale Performanz (99%) wird bei der durchschnittlichen Länge der Testdokumente von 200 Byte (ca. 30 Wörter) erreicht.

Eine Implementierung der N-Gramm Methode in der Kombination mit dem Vektorraum Modell wird auch in der Arbeit von M. Damashek präsentiert und als „Acquaintance technique" bezeichnet. Allerdings wird diese Methode für ähnlichkeitsbasiertes Clustering verwendet, wobei darauf hingewiesen wird, dass der Algorithmus auch für die Sprachidentifikation eingesetzt werden kann (vgl. DAMASHEK, M. 1995a: 4). Wie gut das System die Sprachen identifiziert, lässt sich nicht ohne weiteres feststellen, da entsprechende Experimente in der Arbeit nicht präsentiert werden.

Eine weitere Realisierung des N-Gramm Ansatzes wird in der Arbeit von T. Dunning 1994 vorgestellt. T. Dunning weist darauf hin, dass mit Ausnahme des Tokenisierungsschrittes und der Ad Hoc Ranking Methode für den Vergleich von Sprachmodellen seine Arbeit ähnlich der von W. B. Cavnar & J. M. Trenkle ist (vgl. DUNNING, T. 1994: 6). Die Grundidee der Methode von Dunning besteht in der Berechnung der Wahrscheinlichkeit für das Vorkommen des Teststrings in jeder der vom System unterstützten Sprache (vgl. DUNNING, T. 1994: 6). In der Trainingsphase werden Sprachmodelle mittels Markov-Ketten erzeugt. Anschließend wird in der Erkennungsphase die wahrscheinlichste Sprache für ein gegebenes Testdokument mit Hilfe der Bayes'schen Entscheidungsregel ausgewählt. Das System wird auf einem Sprachkorpus trainiert, der 50 KByte an Trainingsdaten für Englisch und Spanisch enthält. Die erzielte Erkennungsquote beträgt 92% für Testdokumente mit der Länge von 20 Byte (ca. drei Wörter). Dies verbessert sich zu 99,9% in dem Fall, wenn die Sprache eines 500 Byte langen Textes identifiziert wird, was ungefähr 75 Wörtern entspricht (vgl. DUNNING, T. 1994: 16).

2.3 Wortbasierter Ansatz

Beim wortbasierten Ansatz werden einzelne Wörter für die Erstellung von Sprachmodellen verwendet. Die Sprachmodelle sind in diesem Fall Wörterbücher, die eine Art von Wortfrequenzlisten der zu identifizierenden Sprachen darstellen.

In der Literatur sind mehrere Realisierungen des wortbasierten Ansatzes bekannt. Sie unterscheiden sich in erster Linie durch die Art der Wörterbücher. Einige der Wörterbücher enthalten nur hoch frequente Wörter der jeweiligen Sprache (vgl. MARTINO, M. J. & PAULSEN, R. C. 1996; SOUTER, C. et al. 1994; COWIE, J. et al. 1999). Andere Wörterbücher bestehen aus kurzen Wörtern. Als solche gelten die Wörter, die eine maximale Länge nicht überschreiten. Je nach Autor, variiert die maximale Wortlänge zwischen vier und fünf Zeichen (vgl. GREFENSTETTE, G. 1995: 3; PRAGER, J. M. 1999: 4). In einer weiteren Realisierung basiert die Sprachidentifikation auf der Verwendung von Lexika, die Wörter der

geschlossenen Wortklassen wie Adverbien, Artikel, Konjunktionen, Interjektionen, Numeralien, Präpositionen und Pronomina enthalten (vgl. LINS, R. D. & GONÇALVES, P. 2004).
Außerdem unterscheiden sich die in der Literatur beschriebenen Systeme durch die Wörterbuchgröße und die unterschiedlichen Arten von Frequenzen der in den Lexika enthaltenen Wörter.
Eine ausführliche Beschreibung einzelner Realisierungen folgt in den nächsten Abschnitten.

2.3.1 „Frequent word“ Methode

In einigen Realisierungen des wortbasierten Ansatzes erfolgt die Sprachidentifikation auf der Basis von den häufigsten bzw. meist verwendeten Wörtern der jeweiligen Sprache. Diese Methode wird in der Literatur als „frequent word“ (SOUTER, C. et al. 1994) oder „common word“ Technik (MARTINO, M. J. & PAULSEN, R. C. 1996; COWIE, J. et al. 1999) bezeichnet.
Von C. Souter et al. werden für die Sprachidentifikation geordnete Listen von 100 hoch frequenten Wörtern je Sprache verwendet, die aus den Trainingsdaten extrahiert werden. Allerdings ist nicht ersichtlich, ob absolute oder relative Häufigkeiten in die Wortlisten aufgenommen werden. Der Sprachidentifikationsprozess wird von den Autoren wie folgt beschrieben: jedes Wort aus einem eingegebenen Testdokument wird mit den erstellten Frequenzlisten in jeder der neun unterstützten Sprachen verglichen; falls das Wort in einer der Listen gefunden wird, wird der Zähler für die jeweilige Sprache um eins erhöht (vgl. SOUTER, C. et al. 1994: 188). Es wird angenommen, dass das Dokument in der Sprache verfasst ist, die den höchsten Zählerstand hat.
Das oben beschriebene Spracherkennungssystem hat 91% aller Dokumente korrekt identifiziert (vgl. SOUTER, C. et al. 1994: 184). Die Erkennungsquote hängt stark von der Länge des zu identifizierenden Dokuments ab, da mit der wachsenden Textlänge auch die Wahrscheinlichkeit dafür steigt, dass die Wörter des Testdokuments in den Wortlisten vorzufinden sind.

In dem patentierten Sprachidentifikationssystem von M. J. Martino & R. C. Paulsen werden die 100 am häufigsten verwendeten Wörter der zu identifizierenden Sprachen in Wortfrequenztabellen (WFT = Word Frequency Tables) erfasst. Jedes Wort bekommt einen so genannten normalisierten Frequenzwert (NFO = normalized frequency of occurrence), der auf folgende Weise berechnet wird: die relative Häufigkeit jedes in der Tabelle enthaltenen Wortes wird durch die relative Häufigkeit des Wortes auf dem ersten Platz in der Tabelle dividiert (vgl. MARTINO, M. J. & PAULSEN, R. C. 1996: 8). Die Autoren stellen die Behauptung auf, dass der Gebrauch von normalisierten Wortfrequenzen die Identifikation der Sprachen verbessert, in denen gleiche Wörter vorkommen. Das Wort „que“ gehört beispielsweise zu den höchst frequenten sowohl im Französischen als auch im Spanischen, es hat jedoch einen höheren normalisierten Frequenzwert im Spanischen, was folglich heißt, dass alle in einem spanischen Text gefundenen „que“s sich zu einem höheren Gesamtwert im Vergleich zum Französischen summieren.

Ähnlich wie bei C. Souter et al. wird jedes Wort aus einem eingegebenen Testdokument mit den WTFs verglichen. Wird es in einer WFT gefunden, wird sein normalisierter Frequenzwert zu dem Gesamtwert der jeweiligen Sprache addiert. Die Sprache mit dem größten Gesamtwert wird als Dokumentsprache angenommen.

Von J. Cowie et al. werden im Vergleich zu den oben dargestellten Methoden viel längere Wortlisten verwendet: die erzeugten Sprachmodelle beinhalten Frequenzen von den 1000 am häufigsten verwendeten Wörtern in jeder der 34 unterstützten Sprachen. Anhand von Wortfrequenzen der Testdokumente werden Dokumentmodelle erstellt, die daraufhin mit Sprachmodellen verglichen werden. Das wahrscheinlichste Modell wird auf die gleiche Art und Weise wie bei Martino & Paulsen gewählt, nur werden anstatt normalisierter Werte absolute Worthäufigkeiten verwendet. (vgl. COWIE, J. et al. 1999: 4).

2.3.2 „Short word“ Methode

Eine weitere Realisierung des wortbasierten Ansatzes wird „short word“ oder „small word“ Methode genannt. In diesem Fall enthalten Wörterbücher nur kurze Wörter einer Sprache. Zum Beispiel werden bei J. Prager als Kurzwörter Wörter mit einer Länge von vier und weniger Zeichen definiert (vgl. PRAGER, J. M. 1999: 4), bei G. Grefenstette bestehen kurze Wörter aus fünf und weniger Zeichen (vgl. GREFENSTETTE, G. 1995: 3). Ein Beispiel dafür wird in der unten stehenden Tabelle angeführt:

Dan	Dut	Eng	Fre	Ger	Ita	Nor	Por	Spa	Swe
i	de	the	de	der	di	.	de	de	och
af	van	and	la	die	e	og	a	la	i
og	het	to	le	und	il	det	que	que	att
at	een	of		den	che	,	o	el	som
§	en	a	et	in	la	han	e	en	en
til	in	in	des	von	a	i	do	y	r
for	dat	was	les	.	in	er	da	a	p
en	is	his	du	zu	per	"	no	los	det
om	te	that	"	dem	del	p	um	del	av
der	op	I	en	,	un	til	em	se	fr
er	voor	he	un	fr		at	para	por	med
U	met	as	que	mit	non	som	com	las	den
ikke	die	had	a	das	i	var	se	con	till
eller	De	with	qui	des	si	jeg		un	har
som	zijn	it	dans	ist	le	med	os	para	de

Tabelle 2: Liste der hoch frequenten Kurzwörter in zehn Sprachen (GREFENSTETTE, G. 1995: 3)

Die Anwendung der „short word“ Methode wird von J. M. Prager wie folgt begründet: unter hoch frequenten Wörtern einer Sprache sind viele Funktionswörter wie Pronomen, Präpositionen, Artikel, Konjunktionen, Partikel und Hilfswörter. Diese Wörter sind üblicherweise kennzeichnend für die jeweilige Sprache und können somit effektiv für die Sprachidentifikation eingesetzt werden. Da aber nicht immer alle Funktionswortlisten in allen Sprachen verfügbar sind bzw. deren Erstellung aufwendig ist, wird von J. Prager auf die Sprachidentifikation auf der Basis von Funktionswörtern verzichtet. Anstelle dieser Methode wird die „short word“ Technik eingesetzt, da es ziemlich

wahrscheinlich ist, viele der hoch frequenten Funktionswörter unter Kurzwörtern zu finden (vgl. PRAGER, J. M. 1999: 4). Wie auch bei der N-Gramm Methode wird von J. M. Prager das Vektorraum Modell für den Vergleich von Sprachmodellen mit Testdokumenten genutzt. Die ermittelten Frequenzen der Kurzwörter in den Trainings- und Testdokumenten werden als numerische Werte für die Darstellung von Sprach- und Dokumentvektoren verwendet. Eine ausführliche Darstellung des Vektorraum Modells folgt im Unterkapitel 3.2.

In die Kurzwortlisten von G. Grefenstette werden Wörter bis zur maximalen Länge von fünf Zeichen eingetragen, die mehr als drei Mal im Trainingskorpus vorkommen. Die Wörterbücher werden für neun Sprachen erstellt, deren Größe je nach Sprache zwischen 980 und 2750 Kurzwörtern variiert (vgl. GREFENSTETTE, G. 1995: 2). Anhand von ermittelten absoluten Worthäufigkeiten werden Auftretenswahrscheinlichkeiten der Kurzwörter approximiert. Dafür werden zuerst die absoluten Häufigkeiten aller in der Liste enthaltenen Kurzwörter einer einzelnen Sprache summiert. Anschließend wird die absolute Häufigkeit jedes einzelnen Wortes durch diese Summe dividiert.

Beim Vergleich eines eingegebenen Testdokuments mit den erstellten Wörterbüchern werden die Auftretenswahrscheinlichkeiten der übereinstimmenden Wörter summiert und die Sprache mit dem höchsten Summenwert als Dokumentsprache bestimmt.

2.3.3 Geschlossene Wortklassen

Ein weiteres Beispiel für einen wortbasierten Sprachidentifizierer stellt die Arbeit von R. D. Lins & P. Gonçalves 2004 vor. Die Besonderheit des von R. D. Lins und P. Gonçalves entwickelten Algorithmus besteht in der Anwendung von Wörterbüchern, die ausschließlich Wörter der geschlossenen Wortklassen enthalten. Als *geschlossen* werden von den Autoren diejenigen Wortklassen bezeichnet, die endliche selten erweiterbare Wortmengen darstellen, wie Adverbien, Artikel, Konjunktionen, Interjektionen, Numeralien, Präpositionen und Pronomina (vgl. LINS, R. D. & GONÇALVES, P. 2004: 1128).

Der Hauptbaustein des Algorithmus ist die Funktion, die eine Hypothese über die sprachliche Herkunft eines zu identifizierenden Dokumentes liefert. Die Methode verwendet während der Sprachidentifikation Wörterbücher unterschiedlicher Sprachen, jedoch derselben gegebenen Wortklasse, z.B. für die gegebene Wortklasse *Präposition* werden vier Wörterbücher mit jeweils portugiesischen, spanischen, französischen und englischen Präpositionen eingesetzt.
Die Autoren fanden im Laufe der zahlreichen Tests eine für die Identifikation von Portugiesisch, Spanisch, Französisch und Englisch optimale Reihenfolge von geschlossenen Wortklassen, mit denen als Übergabeparameter die oben vorgestellte Methode aufgerufen werden kann. Der Algorithmus wird im Folgenden zusammengefasst:

- Im ersten Schritt werden die ersten *n* Wörter analysiert. Die Sprachidentifikation mit der oben beschriebenen Methode erfolgt unter Verwendung der Wörterbücher der Klasse *Präposition.*
- Je nachdem, welches Ergebnis der erste Schritt geliefert hat, werden unterschiedliche Wörterbücher für die weitere Vorgehensweise verwendet:
    a. Falls das Ergebnis UNKNOWN, N_EXIST (non-existent) oder SPANISCH ist, werden Wörterbücher der Klasse *Adverb* eingesetzt.
    b. Falls das Ergebnis PORTUGUESE oder ENGLISH ist, werden Wörterbücher der Klasse *Pronomen* eingesetzt.
    c. Falls das Ergebnis FRENCH ist, werden Wörterbücher der Klasse *Artikel* eingesetzt.
- Wird die im ersten Schritt gestellte Hypothese durch das Ergebnis des zweiten Schrittes bestätigt, dann terminiert der Algorithmus. Anderenfalls werden die nächsten *m* Wörter betrachtet (vgl. LINS, R. D. & GONÇALVES, P. 2004: 1132)

Der Sprachidentifizierer wurde auf einem Korpus von 600 Texten mit der Länge von 1000 Wörtern gestestet, wobei folgende Ergebnisse erzielt wurden: die Erkennungsquote bei der Identifikation der HTML Seiten betrug mehr als 80% und der Textdokumente – mehr als 90%. Es wäre jedoch interessant zu

erfahren, wie genau der Algorithmus Dokumente mit weniger als 1000 Wörter identifiziert.
Es ist anzunehmen, dass die Erweiterung des Programms um weitere Sprachen zeitaufwendig sein könnte, da die Anzahl der möglichen Reihenfolgen der geschlossenen Klassen mit der wachsenden Anzahl der Sprachen exponentiell wächst.

2.5 Experimentelle Vergleiche unterschiedlicher Ansätze zur Sprachidentifikation

Einige Arbeiten, die sich mit dem Thema der automatischen Sprachidentifikation befassen, präsentieren Ergebnisse der vergleichenden Evaluierungen unterschiedlicher Ansätze. In diesem Zusammenhang sind vor allem die Arbeiten von C. Souter et al. 1994, G. Grefenstette 1995, J. Cowie et al. 1999 und J. Capstick et al. 1999 zu erwähnen. Im Folgenden werden die von den oben genannten Forschern bzw. Forschergruppen ermittelten Ergebnisse dargestellt.
Von der Forschergruppe C. Souter et al. wurden drei Ansätze miteinander verglichen: Sprachidentifikation auf der Basis von unikalen Buchstabenkombinationen (engl. „unique character string identification“), frequenten Wörtern (engl. „frequent word recognition“), sowie Häufigkeiten von Bi- und Tri-Grammen (engl. „bigraph/trigraph based recognition“).
Für die Evaluierung wurden von den Autoren neunsprachige Trainings- und Testkorpora aufgebaut. Bei dem Trainingskorpus wurden 100 KByte an Textdaten pro Sprache verwendet. Der Testkorpus enthielt eine gemischte Menge von Dokumenten der Länge 60 und 420 Zeichen. Die durchgeführten Tests erbrachten folgende Ergebnisse (vgl. SOUTER, C. et al. 1994: 183 ff):
Die Sprachidentifikation auf der Basis von unikalen Buchstabenkombinationen hat die schlechtesten Ergebnisse erzielt, die Erkennungsquote lag in diesem Fall unter 24%. Laut den Autoren waren die eingegebenen Testdokumente nicht lang genug, die Wahrscheinlichkeit für das Auftreten von unikalen Buchstaben-kombinationen war somit eher gering (vgl. SOUTER, C. et al. 1994: 183).

Bei der Bi-Gramm-Methode wurden nur 88% aller Texte korrekt erkannt, wobei die Erkennungsquote von 100% bei der Dokumentlänge von 200 Zeichen erreicht werden konnte. Beim wortbasierten Ansatz wurde ein etwas höherer Anteil (91 %) der Dokumente richtig erkannt. Die beste Erkennungsqualität (94 %) bei kürzeren Texten lieferte die Tri-Gramm Methode, wobei 100-prozentige Genauigkeit bei der Dokumentlänge von 175 Zeichen erzielt werden konnte. Im Allgemeinen wurde bei allen Methoden beobachtet, dass die Verbesserung der Erkennungsrate mit der Vergrößerung der Test-dokumentlänge einherging.

Ein weiterer Vergleich wurde von G. Grefenstette vorgenommen. In seiner Arbeit "Comparing two language identification schemes" wurden die Evaluierungsergebnisse von „Trigram Technique" und „Small Word Technique" einander gegenübergestellt.

Bei der Sprachmodellierung wurden die Tri-Gramme in das Sprachmodell aufgenommen, deren absolute Häufigkeit größer als 100 war. Die Wortlisten bildeten die Wörter, die in den Trainingstexten häufiger als fünf Mal vorgekommen waren. Die auf diese Weise erzeugten Sprachmodelle für neun Sprachen enthielten je nach Sprache zwischen 2550 und 3560 Tri-Gramme. Die Länge der Wortliste betrug zwischen 980 und 2750 Wörter.

Die durchgeführten Tests bestätigten wieder die bessere Performanz der Tri-Gramm Methode bei der Sprachidentifikation von kurzen Testdokumenten (vgl. GREFENSTETTE, G. 1995: 4). Von G. Grefenstette wurde eine Tabelle mit Ergebnissen zusammengestellt, wobei Erkennungsquoten für jede der neun Sprachen einzeln eingetragen wurden. Die Erkennungsquote der Tri-Gramm Methode für die Textlänge von drei bis fünf Wörtern variierte zwischen 86,9% für Spanisch und 97,2% für Deutsch, Englisch und Dänisch (durchschnittliche Quote beträgt 93,7%). Bei der Kurzwort-Methode variierte der prozentuelle Anteil der richtig identifizierten Dokumente zwischen 61,6 % und 97,4% (durchschnittlich 79,9%). Die Kluft vergrößerte sich bei sehr kurzen Texten, die aus einem oder zwei Wörtern bestanden. Bei der Sprachidentifikation längerer Texte (21 und mehr Wörter) wurde von den beiden Methoden eine 99,9%-100% Genauigkeit erzielt (außer Dänisch).

J. Cowie et al. stellten in ihrer Arbeit einen auf dem N-Gram Ansatz basierten Sprachidentifizierer mit einer neuen Klassifikationsmethode vor. Der dem Sprachidentifizierer zugrunde liegende Algorithmus wurde evaluiert, indem seine Performanz mit der von drei anderen Algorithmen: „a simple common words algorithm", „a common words algorithm that uses a notion of distance" und „the Cavnar & Trenkle n-gram algorithm" verglichen wurde. Die ersten zwei Algorithmen realisieren den wortbasierten Ansatz und unterscheiden sich lediglich in der Klassifikationsvorgehensweise.

Für die Evaluierung der Algorithmen wurden Sprachmodelle aus den 1000 häufigsten N-Grammen bzw. Kurzwörtern mit deren relativen Häufigkeiten pro Sprache erzeugt. Die ermittelten Fehlerquoten werden differenziert nach der Länge der getesteten Dokumente in der nachfolgenden Tabelle zusammengefasst:

Länge der Testdokumente (Byte)	Der Algorithmus von Cowie et al.	Der Algorithmus von Cavnar& Trenkle	Der "Distance Common Word" Algorithmus	Der „Common Word" Algorithmus
1000	0,27	3,03	1,83	1,80
500	0,52	3,10	2,48	2,71
100	2,02	4,23	8,20	10,10
50	4,01	6,86	18,83	22,59
20	11,92	16,56	45,97	48,55

Tabelle 3: Fehlerquoten der Algorithmen bei unterschiedlicher Testdokumentlänge (vgl. COWIE, J. et al. 1999: 6)

Bei der Analyse der dargestellten Testergebnisse lässt sich nochmals feststellen, dass die N-Gramm basierte Methode im Vergleich zu der wortbasierten Methode eine bessere Performanz bei den kürzeren Texten erreicht: für Texte der Länge von 20 bis 100 Byte erzielen die Methoden von Cowie et al. und von W. B. Cavnar & J. M. Trenkle eine eindeutig bessere Erkennungsqualität; bei den Textlängen von 500 bis 1000 Byte zeigen alle Methoden eine vergleichbare Performanz.

Interessante Ergebnisse eines Vergleichs der Methode von W. B. Cavnar & J. M. Trenkle mit denen von G. Grefenstette wurden im Bericht von J. Capstick et al. geliefert.

Die Forschergruppe J. Capstick et al. präsentierte MULINEX, ein System für multilinguale Indexierungs-, Navigations- und Editier-Extensionen für das WWW (vgl. http://mulinex.dfki.de/index-d.html, Zugriff 11.09.2005, 10:28 MEZ). Einer der wichtigsten Bausteine des Systems ist der Sprachidentifizierer, der am Anfang des Textbearbeitungsprozesses eingesetzt wird. Weitere Schritte wie Zusammenfassung, thematische Klassifikation und maschinelles Übersetzen bauen auf dem Ergebnis der Sprachidentifikation auf. Als Grundlage für die Entwicklung des Identifizierers diente der Algorithmus von W. B. Cavnar & J. M. Trenkle. Die Auswahl der Methode beruhte auf der Annahme, dass N-Gramme unterschiedlicher Länge die Vorteile verschiedener zur Sprachbestimmung verwendeter Verfahren kombiniert: durch die variierende Länge der N-Gramme werden Häufigkeiten sowohl einzelner Buchstaben, Bi-Gramme, Tri-Gramme, als auch kurzer Wörter herangezogen. Diese Annahme konnte jedoch nicht bestätigt werden (vgl. CAPSTIK, J. et al. 1999: 8). Unten wird eine Tabelle abgebildet, in der Vergleichsresultate der im MULINEX implementierten Methode mit der „Trigram Technique“ und „Small Word Technique“ von G. Grefestette zusammengefasst werden:

	Method	3 to 5 words	6 to 10	11 to 15	16 to 20	21 or more
English	N-Gram	68.75	97.1	99.3	99.7	100
	Trigram	97.2	99.5	99.9	99.9	100
	short words	87.7	97.3	99.8	99.9	100
German	N-Gram	95.1	98.4	99.2	99.9	100
	Trigram	97.2	99.3	99.8	99.9	100
	short words	71.6	89.6	98.2	99.8	100
French	N-Gram	84.9	98	98	98.3	99.5
	Trigram	93	94.5	93.6	99.8	100
	short words	81.8	96	97.2	99.8	100

Tabelle 4: Evaluierungsergebnisse von N-Gramm, Tri-Gramm und „short words“ Methoden (CAPSTIK, J. et al. 1999: 9)

Die durchschnittliche Erkennungsquote der N-Gramm Methode für kurze Textabschnitte (drei bis fünf Wörter) beträgt nur 82,9%. Diese Quote ist wesentlich niedriger als die der Tri-Gramm Methode (95,8%). Ein zufrieden

stellendes Ergebnis für alle Methoden wird erst bei der Sprachbestimmung längerer Texte (21 und mehr Wörter) erzielt.

2.6 Vorteile und Nachteile der N-Gramm und wortbasierten Ansätze

Im Folgenden wird eine Zusammenfassung von Vor- und Nachteilen der N-Gramm- und wortbasierten Ansätze gemacht, die u. a. auf den Ergebnissen der im vorigen Abschnitt dargestellten Vergleichsexperimente beruht.
Die N-Gramm Methode eignet sich besser zur Sprachidentifikation kürzerer Texte als die wortbasierte Methode, wobei innerhalb der N-Gramm Methode die Verwendung von Tri-Gramm Häufigkeiten zu einer besseren Performanz führt, als die Verwendung von Häufigkeiten der N-Gramme unterschiedlicher Länge.
Weitere Stärken bzw. Schwächen der beiden Ansätze können hervorgehoben werden, indem deren Erfolg bei der Identifikation von Sprachen mit schwer zu erkennenden Wortgrenzen (wie z. B. Japanisch, Chinesisch etc.), von stark flektierenden Sprachen (z. B. slawische Sprachen) und bei der Sprachidentifikation von Texten mit seltenem Vokabular verglichen wird. Unter letzteren werden Texte mit stark abweichender Lexik wie Eigennamen und seltenen Wörtern wie Fachbegriffe, Archaismen und Neologismen verstanden. Während die drei aufgezählten Sprach- und Textbesonderheiten keinen entscheidenden Einfluss auf die Erkennungsqualität beim N-Gramm Ansatz haben, sind wortbasierte Sprachidentifizierer nicht anwendbar für Sprachen wie Japanisch, Chinesisch etc., sie erzielen schlechtere Performanz bei Sprachen mit reicher Morphologie und bei Dokumenten mit einem hohen Anteil an seltenen Wörtern. In den letzten zwei Fällen müssen Wörterbücher stark erweitert werden, damit bessere Erkennungsquoten erreicht werden können.
N-Gramm basierte Systeme sind außerdem robust gegen Textfehler (z. B. Fehler, die durch Optical Character Recognition (OCR) Systeme verursacht werden oder Tippfehler in E-Mails oder Forenbeiträgen). Wortbasierte Sprachidentifizierer sind dagegen viel empfindlicher für diese Fehlerart, weil

einzelne falsch erkannte Zeichen im starken Maße die Wortstatistiken beeinträchtigen können. (vgl. CAVNAR, W. B. & TRENKLE, J. M. 1994: 12).
Allerdings hat der wortbasierte Ansatz auch seine Vorteile. Die Stärke dieses Ansatzes besteht vor allem darin, dass sich die Wortlisten im Unterschied zu den N-Gramm-Listen manuell bearbeiten lassen. Durch relativ einfach durchführbare Ergänzungen der Wörterbücher kann die Erkennungsqualität der Sprachidentifizierer wesentlich verbessert werden. Außerdem können die Fehlerquellen leicht identifiziert und eliminiert werden, während N-Gramm-Statistiken kaum manuell zu kontrollieren sind (vgl. LANGER, S. 2002: 8). Natürlich ist bei den manuellen Änderungen der Wörterbücher mit einem erhöhten Trainingsaufwand zu rechnen.

2.7 Kombinierte Ansätze

Von einigen Forschern wurden Sprachidentifikationssysteme entwickelt, die auf der Kombination der N-Gramm und wortbasierten Ansätze basieren. Diese Kombination ermöglicht die Ausnutzung der Vorteile beider Ansätze und führt somit zu einer erhöhten Erkennungsqualität. Im Folgenden werden Arbeiten von J. M. Prager und B. M. Schulze vorgestellt, die die Idee der kombinierten Ansätze bei der Entwicklung ihrer Systeme umgesetzt haben.
J. M. Prager, dessen Arbeit in den vorigen Kapiteln schon mehrmals erwähnt wurde, experimentierte nicht nur mit unterschiedlichen Spracheinheiten zwecks Sprachmodellierung, sondern auch mit deren Kombinationen. Er testete folgende N-Gramm-Wort-Kombinationen:

- Kurze Wörter (der Länge von vier und weniger Zeichen) plus Tri-Gramme,
- Kurze Wörter (der Länge von vier und weniger Zeichen) plus Quad-Gramme,
- Wörter der unbegrenzten Länge plus Quad-Gramme.

In seinem Vektorraum basierten Ansatz stellten die Komponenten von Sprachmodell- und Testdokument-Vektoren Häufigkeiten sowohl von Wörtern als auch von Tri-Grammen bzw. Quad-Grammen dar.

Im Laufe der Tests konnte gezeigt werden, dass die Kombination von N-Grammen und Wörtern im Vergleich zu deren getrennten Anwendung bei der Sprachmodellierung tatsächlich vorteilhafter ist. Aus der Tabelle 5 ist zu sehen, dass kombinierte Methoden vor allem bei der Sprachidentifikation kürzerer Texte bessere Ergebnisse liefern, wobei die beste Performanz bei der Kombination von Wörtern der unbegrenzten Länge und Quad-Grammen erreicht wird.

	Chunk Size (Byte)					
Feature-set	20	50	100	200	500	1000
2-grams	68,8	86,2	93,5	97,7	98,8	100,0
3-grams	79,5	93,0	97,7	99,3	100,0	100,0
4-grams	83,6	94,3	98,2	99,6	99,9	100,0
5-grams	81,4	93,1	97,8	99,4	99,9	99,9
Words	69,7	86,6	94,7	98,1	99,9	100,0
Small words	61,3	81,5	92,1	97,1	99,6	100,0
Small words + 3-grams	*83,8*	*94,9*	*98,5*	*99,7*	*100,0*	*100,0*
Small words + 4-grams	*84,9*	*95,3*	*98,6*	*99,7*	*99,9*	*100,0*
Words + 4-grams	***85,4***	***95,6***	***98,7***	***99,7***	***99,9***	***100,0***

Tabelle 5: Performanz von Linguni differenziert nach Testdokumentlänge und Spracheinheiten (vgl. PRAGER, J. M. 1999: 5)

Ein weiteres Sprachidentifikationssystem, das die Vorteile der Kombination beider Ansätze ausnutzt, wird in der patentierten Arbeit von B. M. Schulze präsentiert.

Die von B. M. Schulze entwickelte Methode stellt eine Kombination der Tri-Gramm und der Kurzwort Methode dar. Nach der Ansicht des Autors ist diese Kombination für die Sprachidentifikation verwandter Sprachen wie z. B. Spanisch und Portugiesisch nützlich, weil diese Sprachen viele ähnliche Tri-Gramme, aber unterschiedliche Funktionswörter haben. In diesem Fall bietet die Methode Vorteile des N-Gamm basierten Ansatzes, und darüber hinaus kann sie anhand der Wortwahrscheinlichkeiten eng verwandte Sprachen voneinander unterscheiden (vgl. SCHULZE, B. M 2000: 4).

Die Sprachmodelle werden anhand von Auftretenswahrscheinlichkeiten der aus den Trainingsdaten gewonnenen Tri-Gramme und Kurzwörter bzw. Funktionswörter aufgebaut. Für jedes im Testdokument gefundene Tri-Gramm bzw. Wort und jedes Sprachmodell wird überprüft, ob das Tri-Gramm bzw. das

Wort im jeweiligen Sprachmodell vorhanden ist. Wenn das der Fall ist, wird die relative Häufigkeit des Tri-Grammes bzw. Wortes zu der kumulierten Häufigkeit des jeweiligen Sprachmodells addiert. Anschließend wird die Sprache des Modells mit dem höchsten kumulierten Wert als die Dokumentsprache vorgeschlagen.
Bei der Systemevaluierung wurde wie auch bei J. M. Prager herausgefunden, dass der kombinierte Ansatz mit der Erkennungsquote von 99,8% die Performanz der Tri-Gramm- (98,8%) und der wortbasierten Methoden (96,4%) übertrifft (vgl. SCHULZE, B. M. 2000: 14).

2.8 Sprachidentifikation auf der Basis von „word shape tokens“

Eine alternative Herangehensweise zum Problem der Sprachidentifkation stellen die Arbeiten von P. Sibun & L. A. Spitz (1994) und P. Sibun & J. C. Reynar (1996) dar.
Das Hauptaugenmerk in der ersten Arbeit wird auf die Sprachidentifikation der Dokumente gelegt, die vorwiegend in Papierform zugänglich sind: zum Beispiel Faxe, Patentanträge, Office Memos etc. (vgl. SIBUN, P. & SPITZ, L. A. 1994: 15). Die Sprachidentifikation solcher Dokumentarten ist eine der Voraussetzungen für deren effektive Weiterbearbeitung mit einem Optical Character Recognition (OCR) System.
Die Idee der Methode basiert auf der Beobachtung, dass Wörter einer Sprache typische Umrissmuster (engl. „word shape tokens“) aufweisen, die für die Sprachidentifikation eingesetzt werden können. Diese spezifischen Wort-Umrissmuster werden aus den gescannten Trainingsdaten gewonnen, indem jedes Zeichen (engl. „character“) des Dokuments einer so genannten Zeichen-Umriss-Kodierung (engl. „character shape codes“) zugeordnet wird. Diese Zuordnung erfolgt anhand der von den Autoren zusammengestellten Tabelle, in der Buchstaben und Zeichen des lateinischen Alphabets mit deren Umriss-Kodierungen aufgeführt sind:

Characters	Character shape codes
bdfhklt#$&%A-Z0-9*ß	A
Ce	E
Çgpqy	G
Àáâèéêîïñôùû	I
J	J
N	N
Amorsuvwxz	X
;:	:
?!	!
”“	’
,	,
.	.
=	=
- ~	-
< > [] () {} \/ |	|
_ (underscore)	_

Tabelle 6: Buchstaben/Zeichen und deren Umriss-Kodierungen
(SIBUN, P. & REYNAR, J.C. 1996: 129)

Ein Beispiel einer solchen Transformation wird in der Abbildung 3 gezeigt:

Character codes

```
Confidence in the international
monetary system was shaky enough be-
fore last week’s action.
```

Character shape codes

AxxAAxxxx ix AAx ixAxxxxAixxxA xxxxAxxg xgxAxx xxx xAxAg xxxxgA Ax-Axxx AxxA xxxA’x xxAixx.

Abbildung 3: Darstellung eines Beispieltextes mit Umriss-Kodierung
(SIBUN, P. & SPITZ, L. A. 1994: 15)

Bei der Analyse aller Wort-Umrissmuster einer Sprache werden charakteristische Muster ausgewählt. Als charakteristisch werden in diesem Fall diejenigen Umrissmuster bezeichnet, die in einer Sprache mehr und in den übrigen weniger häufig sind. Zum Beispiel, AAx ist ein typisches Muster (z.B. für das Wort *the*), das 7% aller Umriss-Repräsentationen in englischen Trainingsdaten beträgt (vgl.

SIBUN, P. & SPITZ, L. A. 1994: 17). Die aus den Trainingsdaten gewonnenen typischen Umrissmuster werden in einer Liste zusammengefasst und ihrer Häufigkeit nach sortiert.

Um ein Testdokument einer Sprache zuordnen zu können, werden seine Umriss-Kodierungen mit den erstellten Listen mittels der linearen Diskriminanzanalyse (LDA) verglichen.

Gegen alle Erwartung erwies sich die Methode beim Testen auf einem Korpus aus 23 Sprachen des romanischen Alphabets als nicht besonders effektiv: bei einigen Sprachen lag der Prozentsatz der korrekt erkannten Dokumente unter 90%, die durchschnittliche Quote betrug ca. 92% (vgl. SIBUN, P. & SPITZ, L. A. 1994: 19).

In der zweiten Arbeit wurde die oben beschriebene Technik für die Sprachidentifikation sowohl von Online Dokumenten als auch von Dokumenten in gedruckter Form, die nach dem Scannen zu Umrissmuster-Repräsentationen konvertiert wurden, verwendet (vgl. SIBUN, P. & REYNAR, J.C. 1996: 1). Bei der Realisierung der Methode wurden einige Änderungen vorgenommen: der Trainingskorpus wurde um drei weitere Sprachen erweitert, LDA wurde durch die Berechnung von Kullback Leibler Distanz ersetzt und anstatt von Wort-Umrissmustern wurden Uni-, Bi- und Tri-Gramm-Umrissmuster eingesetzt. Die erzielte Erkennungsquote variierte zwischen 81,3% bei der Verwendung von Uni-Grammen und 97,3% bei der Verwendung von Tri-Grammen (vgl. SIBUN, P. & REYNAR, J.C. 1996: 8).

Eine vergleichbare Methode wurde von C. L. Tan et al. für die Identifikation von Chinesisch, Englisch, Malaysisch und Tamile implementiert, wobei die durchschnittliche Erkennungsquote von mehr als 90% erreicht wurde (vgl. TAN, C. L. et al. 1999: 5).

3 Klassifikationsverfahren

Aus der Fachliteratur sind mehrere Klassifikationsverfahren bekannt, die in der Erkennungsphase eines Sprachidentifikationsprozesses zum Vergleich von eingegebenen Testdokumenten bzw. deren Modellen mit Sprachmodellen eingesetzt werden können. In diesem Kapitel werden einige dieser Verfahren vorgestellt:

- Ad hoc Ranking (so genannte „out of place“ Methode von CAVNAR, W. B. & TRENKLE, J. M. 1994)
- Das Vektorraum Modell
- Markov-Ketten und Bayes’sche Entscheidungsregel

Diese Methoden werden im Folgenden ausführlich behandelt, weil sie für die Entwicklung des in der vorliegenden Arbeit vorgestellten Sprachidentifikationssystems LangIdent verwendet wurden.

3.1 Ad hoc Ranking (“out of place” Methode)

Die Grundidee des Ad hoc Ranking Verfahrens („out of place“) liegt in der Berechnung der Distanz zwischen dem Modell eines Testdokuments (von Autoren als Dokumentprofil bezeichnet) und Modellen der im Trainingskorpus enthaltenen Sprachen. Die Dokument- und Sprachmodelle stellen Listen von N-Grammen dar, in denen N-Gramme absteigend nach ihrer absoluten Häufigkeit sortiert sind (s. Punkt 2.2.1, S. 11).

Die Methode ermittelt, wie weit ein im Dokumentmodell vorkommendes N-Gramm von seiner Position im jeweiligen Sprachmodell entfernt ist. Wenn ein N-Gramm in den beiden Modellen denselben Rang hat, so ist die Abweichung gleich Null; in anderen Fällen ist die Abweichung der Anzahl der dazwischen liegenden Ränge gleichzusetzen. Falls ein N-Gram in dem Sprachmodell fehlt, wird diesem N-Gramm ein maximaler „*out of place*“ Wert zugewiesen. Alle Abweichungen werden zusammengerechnet, und daraus ergibt sich der Distanzwert. Als wahrscheinlichste Sprache wird anschließend die Sprache

ausgewählt, deren Modell die geringste Distanz zum Dokumentenprofil aufweist. Die folgende Abbildung veranschaulicht die Vorgehensweise bei der Berechnung der Distanzen:

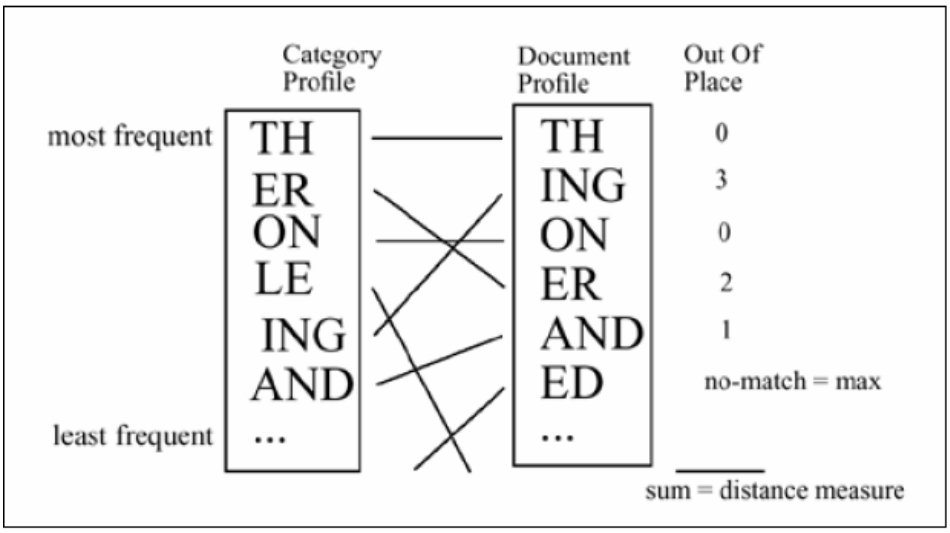


Abbildung 4: Berechnung der *out-of-place* Distanz
(CAVNAR, W. B. & TRENKLE, J. M. 1994: 6)

Zu den Vorteilen der Methode zählen vor allem die einfache Implementierung und ein geringer Speicherbedarf für die Sprachmodelle, da die absoluten Häufigkeiten nach der Erstellung des Modells nicht mehr benötigt werden und nicht mitgespeichert werden müssen. Sie hat aber, wie alle anderen Methoden auch, ihre Schwachstellen. Die „out of place“ Methode funktioniert bei der Identifikation von Sprachen mit ähnlichen Ranglisten nicht immer zuverlässig, weil dieselben N-Gramme in verschiedenen Sprachen zwar den gleichen Rang, aber unterschiedliche absolute Häufigkeiten besitzen können (vgl. SIBUN, P. & REYNAR J. C. 1996: 3). In solchen Fällen wäre die Bewertung anhand von absoluten Häufigkeiten eine bessere Wahl. In der Praxis scheint eine solche Situation aber eher unwahrscheinlich, da die Verteilung der häufigsten N-Gramme doch sehr sprachspezifisch ist und die etwaigen Übereinstimmungen die Spracherkennung nicht wesentlich beeinträchtigen.

3.2 Das Vektorraum Modell

Das Vektorraum Modell (engl. Vector Space Model), eines der meist verbreiteten Modelle in der Information Retrieval Forschung, kann auch für die Sprachidentifikation eingesetzt werden.
Allgemein besteht die Idee dieses Verfahrens in der Berechnung des Ähnlichkeitsmaßes der in einem mehrdimensionalen Raum repräsentierten Vektoren (vgl. http://de.wikipedia.org/wiki/Vektorraummodell, Zugriff 7.06.2005, 9:42 MEZ). Der Grad der Ähnlichkeit wird als Kosinus des Winkels zwischen den Vektoren gemessen. Formal kann das folgendermaßen aufgefasst werden:

$$\cos\alpha = \frac{\vec{a}\cdot\vec{b}}{\left|\vec{a}\right|\cdot\left|\vec{b}\right|} \in [0,1],$$

wobei das Skalarprodukt $\vec{a}\cdot\vec{b}$ und die Euklidische Norm $|\vec{a}|\cdot|\vec{b}|$ wie folgt berechnet werden:

$$\vec{a}\cdot\vec{b} = a_1\cdot b_1 + a_2\cdot b_2 + \ldots + a_n\cdot b_n$$

$$\left|\vec{a}\right|\cdot\left|\vec{b}\right| = \sqrt{a_1^2 + a_2^2 + \ldots + a_n^2}\cdot\sqrt{b_1^2 + b_2^2 + \ldots + b_n^2}$$

Im Vektorraummodell des Information Retrieval werden Dokumente und Anfragen als Vektoren interpretiert. Die Ähnlichkeit zwischen einem Anfragevektor und einem der Dokumentvektoren wird mittels der Kosinusdistanz ermittelt.
Bezogen auf die Sprachidentifikation, wird das zu identifizierende Dokument durch einen Vektor $\vec{d}$ und eine einzelne Sprache F_j aus der Menge der Sprachen $\{F_i\}$ durch den Vektor $\vec{f}_j$ repräsentiert. Um die Sprache F_j, in der das Testdokument geschrieben ist, zu bestimmen, wird nach dem Vektor $\vec{f}_j$

gesucht, das dem Vektor $\vec{d}$ am ähnlichsten ist. Das Ähnlichkeitsmaß ist auch hier der Kosinus des Winkels zwischen den beiden Vektoren (vgl. PRAGER, J. M. 1999: 6):

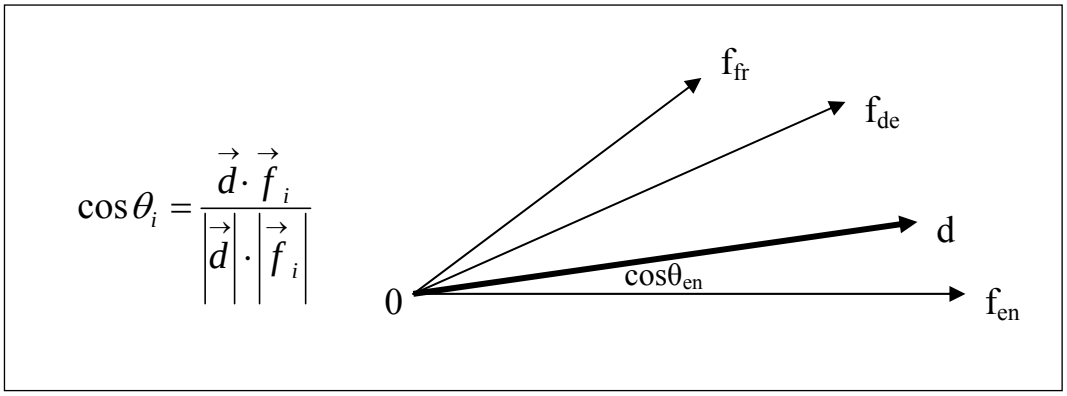


Abbildung 5: Auswahl des ähnlichsten Sprachmodells
(vgl. PRAGER, J. M. 1999: 6)

Um die Formel einsetzen zu können, müssen die numerischen Werte der Vektoren berechnet werden. Die Vektorenwerte sind üblicherweise Frequenzen von N-Grammen oder Wörtern, je nachdem, welcher Ansatz der automatischen Sprachidentifizierung implementiert wird.

Von J. M. Prager werden Vektorenwerte beispielsweise wie folgt berechnet: die absolute Häufigkeit m_i eines N-Gramms bzw. Wortes im Trainingsdokument einer Sprache wird mit der inversen Dokumenthäufigkeit $idf = 1/n_i$ multipliziert, wobei n_i die Anzahl aller Trainingsdokumente darstellt, in denen das N-Gramm bzw. Wort vorkommt (vgl. PRAGER, J. M. 1999: 3).

M. Damashek verwendet relative Häufigkeiten der N-Gramme als Vektorenwerte (vgl. DAMASHEK, M. 1995: 843).

3.3 Klassifikation mittels Bayes'schen Entscheidungsregel und Markov-Ketten

Die Sprachidentifizierung kann unter Einsatz von Methoden aus der Wahrscheinlichkeitstheorie durchgeführt werden (vgl. DUNNING, T. 1994). Die

wahrscheinlichste Sprache eines zu identifizierenden Dokuments wird dabei mit Hilfe des Bayes'schen Theorems und Markov-Ketten ermittelt.

3.3.1 Bayes'sche Entscheidungsregel in der Sprachidentifikation

Als Bayes'sche Entscheidungsregel wird die Auswahl des wahrscheinlichsten Ereignisses aus *A*, *B, C* usw. unter der gegebenen Beobachtung *X* (= das eingetretene Ereignis) bezeichnet. Auf das Sprachidentifizierungsproblem angewandt, sind die Ereignisse einzelne Sprachen und die gegebene Beobachtung ist das zu identifizierende Dokument.
Die Bayes'sche Entscheidungsregel basiert auf dem Bayes'schen Theorem, das wiederum auf dem Konzept der bedingten Wahrscheinlichkeiten aufbaut. Im Folgenden werden diese Grundlagen erläutert.
Unter der bedingten Wahrscheinlichkeit $p(A|B)$ wird die Wahrscheinlichkeit des Ereignisses *A* verstanden, unter der Bedingung, dass Ereignis *B* bereits eingetreten ist. Diese wird wie folgt berechnet (vgl. FRÖTSCHL, B.; LINDSTROT, W. 2004: 117):

$$p(A \mid B) = \frac{p(A \cap B)}{p(B)}$$

Das Bayes'sche Theorem kann anhand der oben angegebenen Formel und der Tatsache, dass ein Durchschnitt zweier Mengen ($A \cap B$) symmetrisch ist, abgeleitet werden. Das Ergebnis der Formelumformung in die multiplikative Beziehung ist (vgl. FRÖTSCHL, B.; LINDSTROT, W. 2004: 118):

$$p(A \cap B) = p(B)\,p(A \mid B) = p(A)\,p(B \mid A)$$

Daraus folgt der Bayes'sche Satz (a. a. O.: 118):

$$p(A \mid B) = \frac{p(A)\,p(B \mid A)}{p(B)}$$

Dieser Satz bringt *a priori* Wahrscheinlichkeit *p(A)* mit *a posteriori* Wahrscheinlichkeit *p(A | B),* also der Wahrscheinlichkeit für das Auftreten des Ereignisses *A* nach dem Auftreten des Ereignisses *B*, in Verbindung.
Auf die Sprachidentifizierung übertragen, ist das Ereignis *A* eine einzelne Sprache und das Ereignis *B* das zu identifizierende Dokument. Die bedingte Wahrscheinlichkeit *p(A | B)* bedeutet in diesem Kontext die Wahrscheinlichkeit für die Sprache *A* bei dem gegebenen Dokument *B*:

$$p(Sprache_A \mid Dokument_1) = \frac{p(Sprache_A)\, p(Dokument_1 \mid Sprache_A)}{p(Dokument_1)}$$

Entsprechend dem Bayes'schen Satz kann nun die wahrscheinlichste Sprache aus einer Menge von Sprachen {$Sprache_A$, $Sprache_B$, $Sprache_C$,…} für das zu identifizierende Dokument bestimmt werden.
Die Wahrscheinlichkeiten auf der rechten Seite der Gleichung können wie folgt behandelt werden (vgl. DUNNING, T. 1994: 8):

- Der Nenner *p (Dokument$_1$)* ist irrelevant, weil er konstant ist. Die Sprache mit der größten Wahrscheinlichkeit kann allein anhand des Zählers *p(Sprache$_A$)p(Dokument$_1$ | Sprache$_A$)* festgestellt werden.
- Die Wahrscheinlichkeit einer Sprache wird in der Praxis oft auf *1 / Anzahl der Sprachen* (=Sprachmodelle des Trainingskorpus) gesetzt.
- Die Wahrscheinlichkeit *p(Dokument$_1$ | Sprache$_A$)* wird mittels Markov-Ketten bestimmt. Die Vorgehensweise dafür wird im nächsten Abschnitt detailliert beschrieben.

3.3.2 Sprachmodellierung mittels Markov-Ketten

Mittels Markov-Ketten können Sprachmodelle {*Sprache$_A$, Sprache$_B$,* …} erstellt werden, die anschließend für die Berechnung der Wahrscheinlichkeiten *p(Dokument$_i$ | Sprache$_j$)* eingesetzt werden. Vor einer detaillierten Beschreibung der Modellerzeugung wird das Konzept von Markov-Ketten kurz angesprochen.

Eine Markov-Kette wird als ein Zufallsprozess definiert, bei dem „die Wahrscheinlichkeit für den Zustand zum Zeitpunkt *t+1* nur von dem Zustand zum Zeitpunkt *t* abhängt (vgl. http://www.matheboard.de/lexikon/Markow-Prozess,definition.htm, Zugriff: 28.03.2005, 15:14 MEZ).

Etwas formaler: eine Markov-Kette definiert eine Zufallsvariable, deren Werte Folgen von Zuständen sind. Die Wahrscheinlichkeitsverteilung der Variable kann der unten stehenden Formel entnommen werden (vgl. DUNNING, T. 1994: 6):

$$p(S) = p(s_1 ... s_n) = p(s_1) \prod_{i=2}^{n} p(s_i \mid s_{i-1}),$$

wobei $S=(s_1 \ldots s_n)$ eine Folge von Zuständen ist. Die Wahrscheinlichkeit $p(S)$ wird durch die Ausgangswahrscheinlichkeitsverteilung $p(s_1)$ und die Übergangswahrscheinlichkeiten $p(s_i|s_{i-1})$ charakterisiert.

Die Übergangswahrscheinlichkeiten $p(s_i|s_{i-1})$ werden üblicherweise mittels einer Matrix (w_{ij}) dargestellt. Der Eintrag w_{ij} in der Matrix liefert die Wahrscheinlichkeit für den Übergang vom momentanen Zustand *i* zum nachfolgenden Zustand *j*.

Übertragen auf die Sprachen kann jeder Buchstabe aus dem Alphabet der jeweiligen Sprache einem Zustand zugeordnet werden. Dadurch kann jedes Wort als eine Folge von Zuständen betrachtet werden. Zum Beispiel, wird das Wort „Text“ zur Folge von Zuständen: $(s_t\ s_e\ s_x\ s_t)$. Anschließend wird jede Sprache durch eine Matrix mit Übergangswahrscheinlichkeiten modelliert. Ein Element w_{ij} aus dieser Matrix trägt die Wahrscheinlichkeit dafür, dass in einem Wort der Buchstabe mit dem Index *j* dem Buchstaben mit dem Index *i* folgt. Ein Ausschnitt aus einer solchen Matrix kann wie folgt aussehen:

	a	**b**	**c**	**d**	**…**
a	$p(s_a\|s_a)$	$p(s_b\|s_a)$	$p(s_c\|s_a)$	$p(s_d\|s_a)$	…
b	$p(s_a\|s_b)$	$p(s_b\|s_b)$	$p(s_c\|s_b)$	$p(s_d\|s_b)$	…
c	$p(s_a\|s_c)$	$p(s_b\|s_c)$	$p(s_c\|s_c)$	$p(s_d\|s_c)$	…
d	$p(s_a\|s_d)$	$p(s_b\|s_d)$	$p(s_c\|s_d)$	$p(s_d\|s_d)$	…
…	…	…	…	…	…

Tabelle 7: Matrix mit Übergangswahrscheinlichkeiten

Es zeigt sich in der Praxis, dass die Matrizen mit Übergangswahrscheinlichkeiten zwischen je zwei Zuständen bzw. Buchstaben für die Modellierung einer Sprache nicht ausreichend sind (vgl. DUNNING, T. 1994: 7). Daher wird an dieser Stelle ein erweitertes Konzept der Markov-Ketten verwendet, so genannte Markov-Ketten *k-ter* Ordnung. Hier hängt die Wahrscheinlichkeit des momentanen Zustands von *k* vorherigen Zuständen ab.
In der folgenden Tabelle sind die Übergangswahrscheinlichkeiten dargestellt, wobei die Wahrscheinlichkeit des Auftretens eines Buchstaben von zwei letzten Buchstaben abhängig ist:

	l	M	n	o	p	q	r	s
de	0,009	0,089	0,287	0,032	0,032	0,032	0,352	0,071
di	0,032	0,032	0,014	0,032	0,032	0,032	0,040	0,008

Tabelle 8: Beispiel einer Übergangswahrscheinlichkeitsmatrix
(vgl. PIOTROWSKI, M. 1997: 3)

Basierend auf dem Konzept von Markov-Ketten *k-ter* Ordnung wird die Wahrscheinlichkeit für das Auftreten einer Zeichenkette *S* in der Sprache *A* wie folgt definiert (vgl. DUNNING, T. 1994: 9):

$$p(S \mid A) = p(s_1 \ldots s_k \mid A) \prod_{i=k+1}^{N} p(s_i \mid s_{i-k} \ldots s_{i-1} \mid A)$$

Die Wahrscheinlichkeit $p(s_1 \ldots s_k \mid A)$ steht in dieser Formel für die Wahrscheinlichkeit des ersten Teilstrings mit der Länge *k* in der Sprache *A*. Der Term $p(s_i \mid s_{i-k} \ldots s_{i-1} \mid A)$ ist die Wahrscheinlichkeit dafür, dass in der Sprache *A* nach dem Teilstring der Länge *k* mit dem Anfang an der Stelle *i-k* der Buchstabe s_i folgt. In der Praxis wird die oben angegebene Formel durch die folgende Formel approximiert, indem alle gleichen Teilstrings zusammen gruppiert werden (vgl. DUNNING, T. 1994: 9):

$$p(S \mid A) = \prod_{w_1 \ldots w_{k+1} \in S} p(w_{k+1} \mid w_1 \ldots w_k \mid A) \cdot T(w_1 \ldots w_{k+1}, S)$$

Um numerische Fehler zu reduzieren, wird der Logarithmus der Wahrscheinlichkeit *p(S|A)* berechnet (vgl. DUNNING, T. 1994: 9):

$$\log\ p(S \mid A) = \sum_{w_1 \ldots w_{k+1} \in S} T(w_1 \ldots w_{k+1}, S) \log\ p(w_{k+1} \mid w_1 \ldots w_k \mid A)$$

Um die Wahrscheinlichkeit *p(S|A)* berechnen zu können, muss vorher eine Matrix mit den Übergangswahrscheinlichkeiten (= Sprachmodell) für jede Sprache aus dem Trainingskorpus erstellt werden. Für die Berechnung der Übergangswahrscheinlichkeiten kann der so genannte *„maximum likelihood estimator“* eingesetzt werden (vgl. DUNNING, T. 1994: 10):

$$p(w_{k+1} \mid w_1 \ldots w_k \mid A) = \frac{T(w_1 \ldots w_{k+1}, T_A)}{T(w_1 \ldots w_k, T_A)}$$

Hier wird die absolute Häufigkeit einer Zeichenkette mit der Länge *k+1* durch die absolute Häufigkeit derselben Zeichenkette, jedoch ohne den letzten Buchstaben, dividiert. S_A legt dabei fest, dass für die Ausrechnung der Wahrscheinlichkeiten die Trainingsdokumente der Sprache *A* verwendet werden müssen.

Bei der Verwendung der oben aufgeführten Formel kann jedoch ein Problem auftreten, falls der Trainingskorpus relativ klein ist. Sollte ein Teilstring in dem Trainingset aller Sprachen nicht vorhanden, aber in dem Testdokument vorzufinden sein, dann degradiert die Auftretenswahrscheinlichkeit des Teststrings in jeder Sprache zu Null: *p(S|A)=0*. Oder, sollte ein seltener Teilstring, das normalerweise in mehreren Sprachen vorkommt, nur in einem der erzeugten Sprachmodelle vorhanden sein, dann wird jedes Testdokument, das diesen Teilstring enthält, immer nur dieser Sprache zugeordnet. Die Wahrscheinlichkeiten für alle anderen Sprachmodelle wird dann gleich 0 gesetzt, was zu einer falschen Klassifikation führen kann. In der Praxis treten solche Verzerrungen oft auf (vgl. DUNNING, T. 1994: 10). Eine Abhilfe bietet folgende Schätzung:

$$\hat{p}(w_{k+1} \mid w_1 \ldots w_k \mid A) = \frac{T(w_1 \ldots w_{k+1}, T_A) + 1}{T(w_1 \ldots w_k, T_A) + m},$$

wobei *m* die Anzahl der Buchstaben des jeweiligen Alphabets darstellt. Die angegebene Formel wird nicht weiter erläutert, weil das den Rahmen der vorliegenden Arbeit sprengen würde. Interessierte Leser können in der Arbeit von T. Dunning eine Herleitung der Formel finden.

Die berechneten Übergangswahrscheinlichkeiten werden dann für die Berechnung von *p(S|A)* bzw. *log p(S|A)* verwendet, die anschließend in der Bayes'schen Formel für die Auswahl der wahrscheinlichsten Sprache eingesetzt wird:

$$p(A \mid S) = \frac{p(A)\, p(S \mid A)}{p(S)}$$

Für einen Vergleich zwischen Wahrscheinlichkeiten für jede Sprache ist der Nenner *p(S)* irrelevant, da er konstant ist. Aus diesem Grund ist alleine das Produkt *p(A)p(S|A)* für die Auswahl der wahrscheinlichsten Sprache maßgebend. Die Wahrscheinlichkeit *p(A)* wird in der Praxis oft auch als eine Konstante gesehen und auf *1/Anzahl aller Sprachen* (im Trainingskorpus) gesetzt. Sollte also ein Trainingskorpus aus 10 Sprachen bestehen, so ist die Wahrscheinlichkeit jedes Sprachmodells gleich hoch: 1/10=0,1. Bei einer konstanten Wahrscheinlichkeit *p(A)* wird der Text der Sprache zugeordnet, deren *p(S|A)*- bzw. *log p(S|A)*-Wert maximal ist.

4 Sprachidentifikation multilingualer Texte

Eine besondere Herausforderung für ein Sprachidentifikationssystem stellen Dokumente dar, die in zwei oder mehreren Sprachen verfasst sind.

In der großen Anzahl der wissenschaftlichen Arbeiten zum Thema Sprachidentifikation gibt es erstaunlich wenige, die über Methoden zur Sprachidentifikation multilingualer Dokumente berichten. Im Folgenden werden zwei das Thema behandelnde Arbeiten von J. Prager und M. J. Martino & R. Ch. Paulsen vorgestellt.

J. M. Prager beschreibt in seiner Arbeit einen auf dem Vectorraum Modell basierten Algorithmus (s. Punkt 2.2.2, S.14), der für die Sprachidentifikation von bilingualen Texten eingesetzt werden kann. Obwohl dieser Algorithmus für die Identifikation von bilingualen Dokumenten entwickelt wurde, lässt er sich auch für die Sprachidentifikation von mehrsprachigen Texten anwenden, für die jedoch bekannt sein soll, aus wie vielen Sprachen sie bestehen (vgl. PRAGER, J. M. 1999: 9).

Die Aufgabe des vom Autor vorgestellten Algorithmus ist, zwei wahrscheinlichste Sprachen F_i und F_j zu finden, aus denen das zu identifizierende bilinguale Dokument D besteht. Um dies zu erreichen, werden alle möglichen Sprachenpaare mittels einer Bewertungsfunktion miteinander verglichen, und das beste Paar als Lösung vorgeschlagen. Die Bewertungsfunktion wird vom Autor wie folgt realisiert:

Bei der Bewertung von zwei Sprachen F_i und F_j bezüglich des Dokuments D wird nach einer so genannten „Mischsprache“ K bestehend aus F_i und F_j gesucht, deren Vektor k den kleinsten Winkel zum Dokumentvektor d hat. Hierbei wird der Vektor der Mischsprache auf folgende Weise definiert:

$$k = \alpha f_i + (1-\alpha) f_j$$

Der Faktor α liegt im Intervall von 0 bis 1 und definiert die Anteile der Sprachen F_i und F_j in der Mischsprache K. Die Aufgabe wird also auf die Suche nach dem

Faktor α reduziert, mit dem die zum Dokument *D* ähnlichste Sprache definiert wird. J. M. Prager leitete für diese Suchaufgabe folgende Lösung her:

$$\alpha = \frac{f_i d - f_j d \cdot f_i f_j}{(1 - f_i f_j) \cdot (f_i d + f_j d)}$$

Ausgehend von dem berechneten Faktor α kann einfach der Vektor *k* und der gesuchte Winkel zwischen den Vektoren *k* und *d* bestimmt werden. Der Winkel zwischen den Vektoren bewertet die Wahrscheinlichkeit dafür, dass das Dokument *D* in Sprachen F_i und F_j verfasst ist.

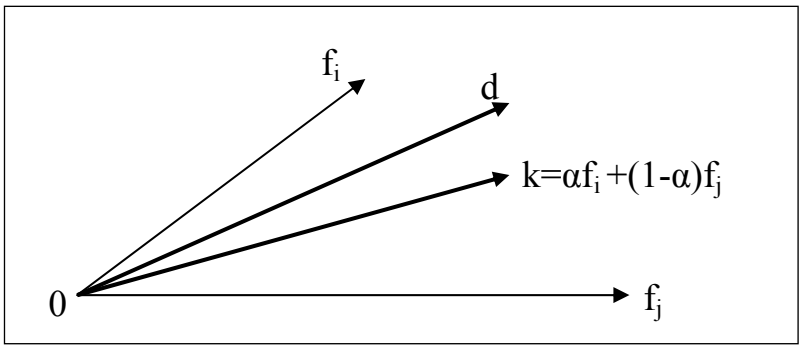


Abbildung 6: Graphische Darstellung der Vektoren f_j, f_j, *k* und *d*
(vgl. PRAGER J. M. 1999: 9)

Der vorgestellte Ansatz lässt sich für Dokumente, die aus drei Sprachen und mehr bestehen, erweitern. Für dreisprachige Dokumente wird *k* wie folgt definiert:

$$k = \alpha f_i + \gamma f_j + (1-\alpha-\gamma) f_l$$

wobei f_l den Vektor einer dritten Sprache F_l darstellt.

Wie effektiv die oben vorgestellte Methode für die Sprachidentifikation von multilingualen Texten ist, lässt sich schwer beurteilen, da von J. M. Prager keine Testergebnisse dokumentiert werden.

M. J. Martino & R. C. Paulsen haben einen Algorithmus zum Patent angemeldet, der speziell für die Sprachidentifikation multilingualer Dokumente entwickelt wurde.

Der Algorithmus soll die Stellen innerhalb eines Dokuments bestimmen, wo ein Sprachwechsel (engl. natural language shift) stattfindet. Dafür wird ein Textintervall der Länge *n* Wörter benutzt, das immer um ein Wort nach rechts verschoben wird, bis das Ende des Dokuments erreicht wird.

Der Sprachmodellierung liegt der wortbasierter Ansatz zu Grunde. Für jede Sprache werden die häufigsten Wörter mit ihrer relativen Häufigkeit in die Wortliste aufgenommen. Die Anzahl der Wörter in der Worttabelle kann von Sprache zu Sprache variieren.

Der Algorithmus beinhaltet folgende wichtige Schritte (vgl. MARTINO, M. J. & PAULSEN, C. 1999: ff):

- Zuerst wird die Länge n des Textintervalls bestimmt, das Wort für Wort durch das ganze Testdokument bewegt wird. Für die Bestimmung der Intervalllänge ist die Sprachmodellgröße von entscheidender Bedeutung, und zwar nicht die Anzahl der Wörter, sondern ihre kumulative Häufigkeit. Wenn die kumulative Häufigkeit der Wörter in einem Sprachmodell etwa 25% beträgt, würde man erwarten, dass jedes vierte Wort in einem Dokument, das in der jeweiligen Sprache verfasst wurde, in dem Sprachmodell zu finden ist. Ist die Länge des Textintervalls kleiner als vier, kann die Sprache des Dokuments in dem Intervall womöglich nicht bestimmt werden.
- In jedem Textintervall wird die Dokumentsprache unter Anwendung der wortbasierten Methode bestimmt. Dafür wird in den Wortlisten aller Sprachen nach den im jeweiligen Intervall enthaltenen Wörtern gesucht. Die Wahrscheinlichkeit für jede Sprache wird bestimmt, indem die Anzahl der in der Wortliste der jeweiligen Sprache gefundenen Wörter durch die Länge des Textintervalls dividiert wird. Die Sprache mit dem höchsten Wahrscheinlichkeitswert wird als aktuelle Dokumentsprache gewählt. Weisen die Wortlisten verschiedener Sprachmodelle unterschiedliche kumulative Häufigkeiten auf, könnte eine Normalisierung notwendig sein.
- wenn die Wörter, die in mehreren Sprachen vorkommen, aus den Wortlisten eliminiert werden, wird in der Regel nur bei einer Sprache der Wahrscheinlichkeitswert größer Null sein, es sei denn, es findet in diesem Textabschnitt ein Sprachwechsel statt.

- In dem Textfenster, wo zwei Sprachen vergleichbare Wahrscheinlichkeitswerte erhalten oder eine andere Sprache als aktuelle Dokumentsprache ausgewählt wird, wird ein Sprachwechsel angenommen. Dieser Abschnitt wird deshalb einer syntaktischen Analyse unterzogen, um die genaue Grenze zwischen Textabschnitten in den beiden Sprachen zu bestimmen: es wird nach den Zeichen bzw. Merkmalen gesucht, die üblicherweise einen Übergang markieren, z.B. ein Zeilenumbruch, Satzzeichen wie Punkt, Komma, Doppelpunkt oder Anführungszeichen sowie Großschreibung, Schriftartwechsel etc.

In der nachfolgenden Abbildung wird eine schematische Darstellung eines in drei Sprachen verfassten Dokuments mit markierten Textfenstern und markierten Stellen des Sprachwechsels gegeben.

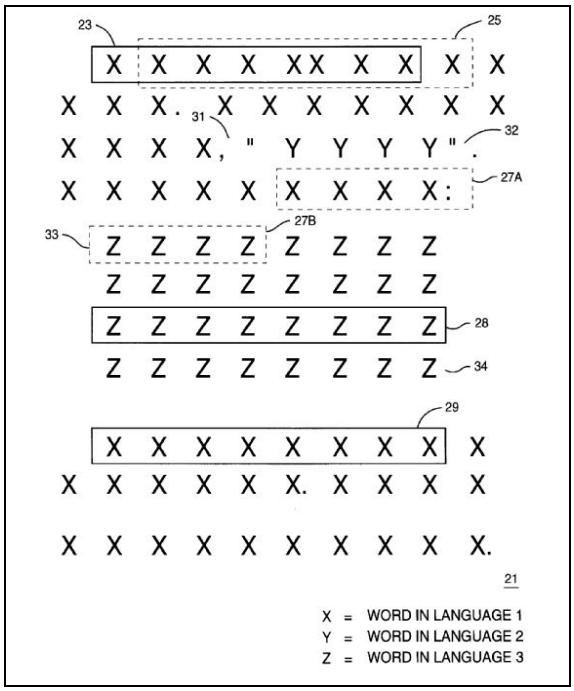


Abbildung 7: Schematische Darstellung eines in drei Sprachen verfassten Dokuments (MARTINO, M. J. & PAULSEN, C. 1999, Fig. 1)

Die Autoren erwähnen, dass mit der von ihnen entwickelten Methode die Position des Sprachwechsels auf wenige Wörter genau bestimmt werden kann. Allerdings werden in der Arbeit keine weiteren Evaluierungsergebnisse angeführt, was eine Beurteilung der Systemeffektivität nicht möglich macht.

5 Werkzeuge zur Sprachidentifikation

Dieser Abschnitt liefert einen Überblick über bereits entwickelte Systeme zur Sprachidentifikation von Textdokumenten. Aus der großen Anzahl der im Internet präsentierten Sprachidentifikationstools werden im Weiteren Freeware oder online zugängliche Werkzeuge beschrieben.

Bei der Untersuchung wurde auf die in den Systemen implementierten Methoden, sowie auf deren Eigenschaften wie die Anzahl der zu identifizierenden Sprachen und die Effektivität der Systeme bei unterschiedlicher Textlänge geachtet. Die Untersuchungsergebnisse werden in der unten stehenden Tabelle zusammengefasst. Allerdings waren die im Internet verfügbaren Informationen nicht immer vollständig, so dass einige Einträge in der Tabelle fehlen.

	Name des Werkzeugs	Methode	Anzahl der Sprachen	Free-ware
1	TextCat[1]	N-Gramm (nach Cavnar & Trenkle 1994)	69	Ja
2	Unknown Language Identification[2]	N-Gram (Algorithmus von M. Damashek)	66	Nein
3	Language Identifier by PetaMem[3]	Wortbasiert, N-Gramm	17	Nein
4	Eidetica[4]	Unikale Buchstabenkombinationen	70	Nein
5	Languid[5]	N-Gram	72	Nein
6	Lingua::Identify[6]	“short word” Methode Präfix Analyse, Suffix Analyse, N-Gramm	26	Ja
7	mguesser[7]	N-Gramm	47	Ja
8	Lingua::Ident[8]	N-Gramm, Markov Ketten, Bayes (nach Dunning 1996)	6	Ja
9	SILC Alis (*Système d'Identification de la Langue et du Codage*)[9]		28	Nein
10	Xerox MLTT Language Identifier[10]		47	Nein
11	Talenknobbel[11]		12	Nein
12	Lextec[12]		260	Ja
13	Polyglot 3000[13]		406	Ja

Tabelle 9: Sprachidentifikationswerkzeuge

[1] http://odur.let.rug.nl/~vannoord/TextCat/ (Zugriff 14.04.2005, 15:38 MEZ)
[2] http://complingone.georgetown.edu/~langid/ (Zugriff 15.04.2005, 17:10 MEZ)
[3] http://nlp.petamem.com/langident.cgi (Zugriff 14.04.2005, 17:15 MEZ)
[4] http://www.eidetica.com/services/guesser (Zugriff 14.04.2005, 17:44 MEZ)
[5] http://languid.cantbedone.org/ (Zugriff 15.04.2005, 12:26 MEZ)
[6] http://search.cpan.org/~cog/Lingua-Identify/lib/Lingua/Identify.pm (Zugriff 15.04.2005, 15:45 MEZ)
[7] http://www.mnogosearch.org/guesser/ (Zugriff 15.04.2005, 16:00 MEZ)
[8] http://search.cpan.org/~mpiotr/Lingua-Ident/Ident.pm (Zugriff 15.04.2005, 15:30 MEZ)
[9] http://rali.iro.umontreal.ca/ (Zugriff 14.04.2005, 16:10 MEZ)
[10] http://www.xrce.xerox.com/competencies/content-analysis/tools/guesser (Zugriff 14.04.2005, 16:21 MEZ)
[11] http://www.fuzzums.nl/talenknobbel/ (Zugriff 14.04.2005, 17:36 MEZ)
[12] http://www.languageidentifier.com/ (Zugriff 14.04.2005, 16:55 MEZ)
[13] http://www.emulator3000.emuita.it/p3.htm (Zugriff 20.05.2005, 19:00 MEZ)

Einige der in der Tabelle 9 aufgeführten Tools basieren auf den wohlbekannten Verfahren, die in den vorherigen Kapiteln dieser Arbeit vorgestellt wurden: das von G. van Noord entwickelte Programm *TextCat* (1) und von A. Barkov programmierte *mguesser* (7) basieren auf der N-Gramm Methode, die in der Arbeit von W. B. Cavnar & J. M. Trenkle vorgestellt wurde; *Unknown Language Identification* (2) von S. Huffman ist auf dem von M. Damashek entwickelten Algorithmus aufgebaut; *Lingua::Ident* (8) von Michael Piotrowski implementiert die statistische Methode von T. Dunning.

Einige der untersuchten Systeme implementieren mehrere Methoden oder deren Kombinationen. Das Sprachidentifikationssystem von der Firma *PetaMem* (3) kann durch die Auswahl folgender Sprachidentifikationsmethoden gesteuert werden: wörterbuchbasierte Methode (`Dict`), N-Gramm basierte Methode (`NGram`), urheberrechtlich geschützte Methode von PetaMem (`NVect`), die eine Generalisierung der N-Gramm Methode darstellt. Die Entwickler behaupten, dass `NVect` bessere Ergebnisse als die N-Gramm Methode (`NGram`) bei kürzeren Texten liefert, jedoch einen erhöhten Rechenzeitbedarf aufweist und nicht für lange Texte geeignet ist (vgl. Hilfeseite von http://nlp.petamem.com/de/langident.cgi, Zugriff 20.05.2005, 21:05 MEZ). Die vierte von PetaMem implementierte Methode heißt `Smart` und erlaubt den Wechsel zu derjenigen Erkennungsmethode, welche für einen gegebenen Text am besten geeignet ist.

Der von J. Castro programmierte Sprachidentifizierer *Lingua::Identify* (4) wird auf vier Methoden aufgebaut: “short word” Methode, N-Gramm Methode, Präfix Analyse und Suffix Analyse. Die Auswahl und Relevanz der Methoden können vom Nutzer bestimmt werden. Darüber hinaus können Sprachen, die vom System identifiziert werden müssen, vom Nutzer aktiviert oder deaktiviert werden.

Die Anzahl der zu identifizierenden Sprachen bei den untersuchten Werkzeugen variiert zwischen sechs Sprachen bei *Lingua::Ident* und 406 bei *Polyglot 3000*.

Viele der untersuchten Sprachidentifizierer bestimmen nicht nur die Sprache von Textdokumenten, sondern auch deren Zeichenkodierungen: *SILC Alis*, *Xerox MLTT Language Identifier*, *Language Identifier by PetaMem*, *Lextec*, *mguesser*.

Einige der Tools unterstützten bei einzelnen Sprachen nur bestimmte Zeichenkodierungen, zum Beispiel, bei *TextCat* trifft das auf Arabisch,

Belarussisch, Bulgarisch, Chinesisch, Japanisch, Russisch etc. zu. Die Sprache eines Textdokuments, das in einer vom System nicht unterstützten Zeichenkodierung verfasst ist, wird nicht identifiziert.

Die Darstellung der Ergebnisse beschränkt sich bei den meisten Systemen auf die einfache Ausgabe der identifizierten Sprache, sowie, falls vorgesehen, auch der Zeichenkodierung des Textdokumentes. Einzelne Systeme liefern jedoch eine Liste von mehreren Sprachen geordnet nach deren Wahrscheinlichkeitswerten, zum Beispiel, bei *Unknown Language Identification* (2) werden die wahrscheinlichste Sprache und die nächsten drei Möglichkeiten ausgegeben:

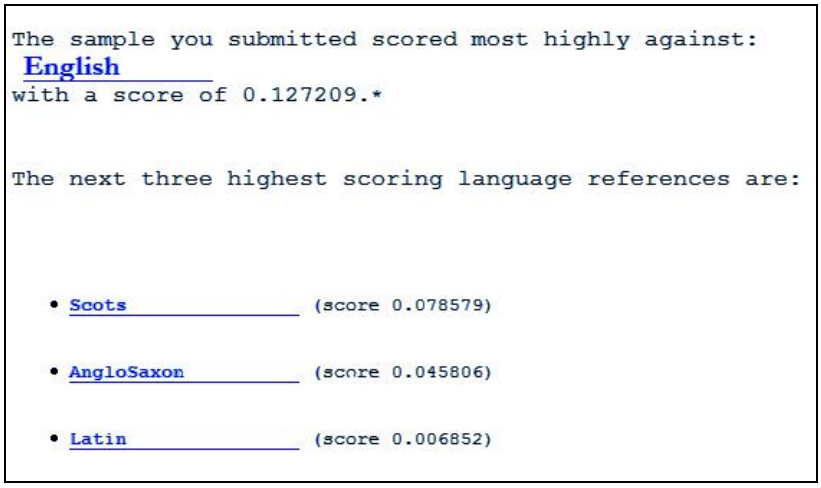


Abbildung 8: Ausgabe von *Unknown Language Identification* [14]

Ausgehend von den im Internet vorhandenen Beschreibungen sollten die hier vorgestellten Sprachidentifizierer schnell und zuverlässig die von ihnen unterstützten Sprachen bestimmen. Von den Autoren der vorliegenden Arbeit konnte diese Aussage jedoch nicht überprüft werden, da die Evaluierung mehrerer Systeme in erster Linie sehr zeitaufwendig ist.

Der Entwickler des Sprachidentifizierers *Polyglot 3000* versichert, dass vom System die Sprachen sowohl einzelner Sätze und Phrasen als auch eines einzelnen Wortes identifiziert werden können (vgl. http://www.emulator3000.emuita.it/p3.htm, Zugriff 20.05.2005, 19:00 MEZ). Bei einigen Sprachidentifizierern wird die minimale Länge der Eingabetexte

[14] http://complingone.georgetown.edu/cgi-bin/langid/cgi_form1.cgi (Zugriff 15.04.2005, 17:15 MEZ)

angegeben: bei *Xerox MLTT Language Identifier* wird die minimale Länge der Testdokumente von fünf Wörtern empfohlen, bei *Lextec* dagegen sollten zu identifizierende Dokumente nicht kürzer als 200 Zeichen sein.

Keiner der oben genannten Sprachidentifikationssysteme eignet sich für Sprachbestimmung von bi- und multilingualen Texten.

Der einzige Identifizierer, der Sprachen von multilingualen Dokumenten und deren Zeichenkodierungen erkennen kann, ist die kommerzielle Software *Rosette Language Identifier*[15]. Da zu diesem System nur eine Demo Version vorliegt und keine Möglichkeit zum Testen besteht, kann hier nichts Konkreteres über das System berichtet werden. Was von dem Hersteller als Demo bezeichnet wird, sind in Wirklichkeit nur ein paar Abbildungen. Es wird nicht erwähnt, welcher Ansatz der Sprachidentifizierung zu Grunde liegt und es ist unklar, wie lang der anderssprachige Text sein muss, um als solcher erkannt zu werden. Der Text in der Abbildung besteht aus drei Absätzen, jeder in einer anderen Sprache. Es ist also möglich, dass der Sprachwechsel innerhalb der Absätze nicht erkannt werden kann.

[15] http://www.basistech.com/products/text-processing/euclid.html (Zugriff 14.04.2005, 16:48 MEZ)

II Entwicklung des Sprachidentifikationssystems LangIdent

Im zweiten Teil der vorliegenden Arbeit werden die wichtigsten Phasen der Entwicklung des Sprachidentifikationssystems LangIdent vorgestellt. Nach der Zielsetzung und der Anforderungsanalyse werden die Kernkomponenten des Systems sowie deren Realisierung beschrieben. Anschließend werden die Evaluierungsergebnisse vorgestellt, die von LangIdent bei der Sprachidentifikation von mono- und multilingualen Dokumenten erzielt wurden.

6 Phasen der Entwicklung von LangIdent

Der Prozess der Softwareentwicklung kann in folgende Kernphasen aufgeteilt werden (vgl. http://de.wikipedia.org/wiki/Softwareentwicklungsprozess, Zugriff 3.07.2005, 21:57 MEZ):

- Anforderungsanalyse
- Systementwurf (Systemaufbau)
- Implementierung
- Tests

Obwohl das Sprachidentifizierungssystem LangIdent keinem großen Softwareprojekt gleichzusetzen ist, erfolgt seine Entwicklung nach der oben beschriebenen Phasenaufteilung.

In der ersten Phase werden Anforderungen an das zu entwickelnde System erarbeitet und anschließend ausführlich dokumentiert (Kapitel 7).

In der nächsten Phase wird ein Modell im Hinblick auf das zu entwickelnde System und die gestellten Anforderungen beschrieben. Das Modell stellt eine Abstraktion des Systems dar, die Systemkomponenten, deren zeitliches Verhalten und Kommunikation definiert (Kapitel 8, 9).

Die Umsetzung des entworfenen Systemmodells in eine Programmiersprache erfolgt in der nachfolgenden Implementierungsphase (Kapitel 10).

In der abschließenden Phase wird das ganze System anhand der gestellten Anforderungen getestet. Zum einen wird nach Unterschieden zwischen dem erwarteten Systemverhalten und dem wirklichen Systemverhalten gesucht, zum anderen wird die Effektivität des Systems bei der Sprachidentifikation überprüft (Kapitel 13).

7 Ziel und Anforderungen

In einem der vorherigen Kapitel wurden eine Vielzahl von Sprachidentifikationswerkzeugen vorgestellt. Es handelt sich um kommerzielle Software und Open Source Programme, eigenständige Applikationen und Bestandteile größerer Software-Pakete. Sie unterscheiden sich in der Anzahl der Sprachen, die identifiziert werden können, in dem Funktionsumfang und auch in ihrer Performanz.

Bei den beschriebenen Softwaretools fehlen jedoch Funktionalitäten, die in der Lehre und Forschung nützlich sein können. Zum einen fehlt die Möglichkeit, sich die erstellten Sprachmodelle anzusehen, gar sie zu bearbeiten. Zum anderen ist es unmöglich, zu bestimmen, wie das Sprachmodell erstellt werden soll. Eine Auswahl unterschiedlicher Klassifikationsmethoden ist ebenfalls nicht gegeben. Auch bei der Erkennung mehrerer Sprachen in einem Dokument scheitern alle getesteten Werkzeuge.

Im Gegensatz dazu soll das im Rahmen der vorliegenden Arbeit zu entwickelnde Programm alle Sprachen eines elektronischen Textdokuments identifizieren. Außerdem solle es dem Nutzer ermöglichen, bei der Sprachmodellerstellung aktiv mitzuwirken und selbst zu entscheiden, welche Spracheinheiten in das Modell mit aufgenommen werden sollen. Aufgrund der Tatsache, dass die Sprachmodelle dem Nutzer nicht verborgen bleiben, können die (eventuell falschen) Ergebnisse besser analysiert und nachvollzogen werden.

Die an das System gestellten Anforderungen werden im Folgenden zusammengefasst.

Bei der *Sprachmodellierung* sollen vom System folgende Funktionalitäten gewährleistet werden:

a. Das System soll jederzeit um weitere Sprachen erweitert werden können.
b. Es soll dem Nutzer möglich sein, die erstellten Sprachmodelle in externen Dateien zu speichern und sie nach Bedarf in den Arbeitsbereich zu laden.
c. Bei der Erstellung der Sprachmodelle soll es die Möglichkeit geben, auszuwählen, welche Tri-Gramme und Wörter in das Modell aufgenommen werden sollen.

d. Das System soll dem Nutzer ermöglichen, sich die Sprachmodelle zu einem späteren Zeitpunkt anzusehen und die Wortlisten nachträglich intellektuell zu bearbeiten.

In Bezug auf den Prozess der *Sprachidentifikation* sollen dem Nutzer folgende Optionen zur Auswahl gestellt werden:

a. Identifikation entweder der primären Sprache eines Testdokuments oder Identifikation aller Sprachen, in denen das Testdokument verfasst ist.
b. Im Fall der Identifikation der primären Sprache eines Testdokuments soll es möglich sein, zwischen mehreren Klassifikationsmethoden zu wählen: Ad Hoc Ranking, Vektorraum Modell, Bayes'sche Entscheidungsregel oder wortbasierte Methode.

Außerdem soll das System über eine benutzerfreundliche Oberfläche verfügen, die eine leichte Navigation und Anwendung des Programms auch für ungeübte Nutzer ermöglicht.

Allerdings richtet sich das zu entwickelnde System vor allem an fortgeschrittene Anwender, die sich bereits mit dem Thema Sprachidentifikation befasst haben. Es soll dem Benutzer ermöglichen, die Effizienz unterschiedlicher Algorithmen zu vergleichen, die optimalen Einstellungen für die Sprachmodellbildung herauszufinden, die für verschiedene Methoden durchaus unterschiedlich sein können, die Zusammenhänge zwischen der Qualität des Trainingsmaterials und den Testergebnissen zu untersuchen u. v. m.

8 Systemaufbau

Dieses Kapitel liefert einen Überblick über den Aufbau des von den Autoren der vorliegenden Arbeit entwickelten Systems zur Sprachidentifikation. Es wird an dieser Stelle insbesondere auf Systemkomponente, deren Zusammenhänge sowie den Datenverarbeitungsprozess eingegangen (Abb. 9).

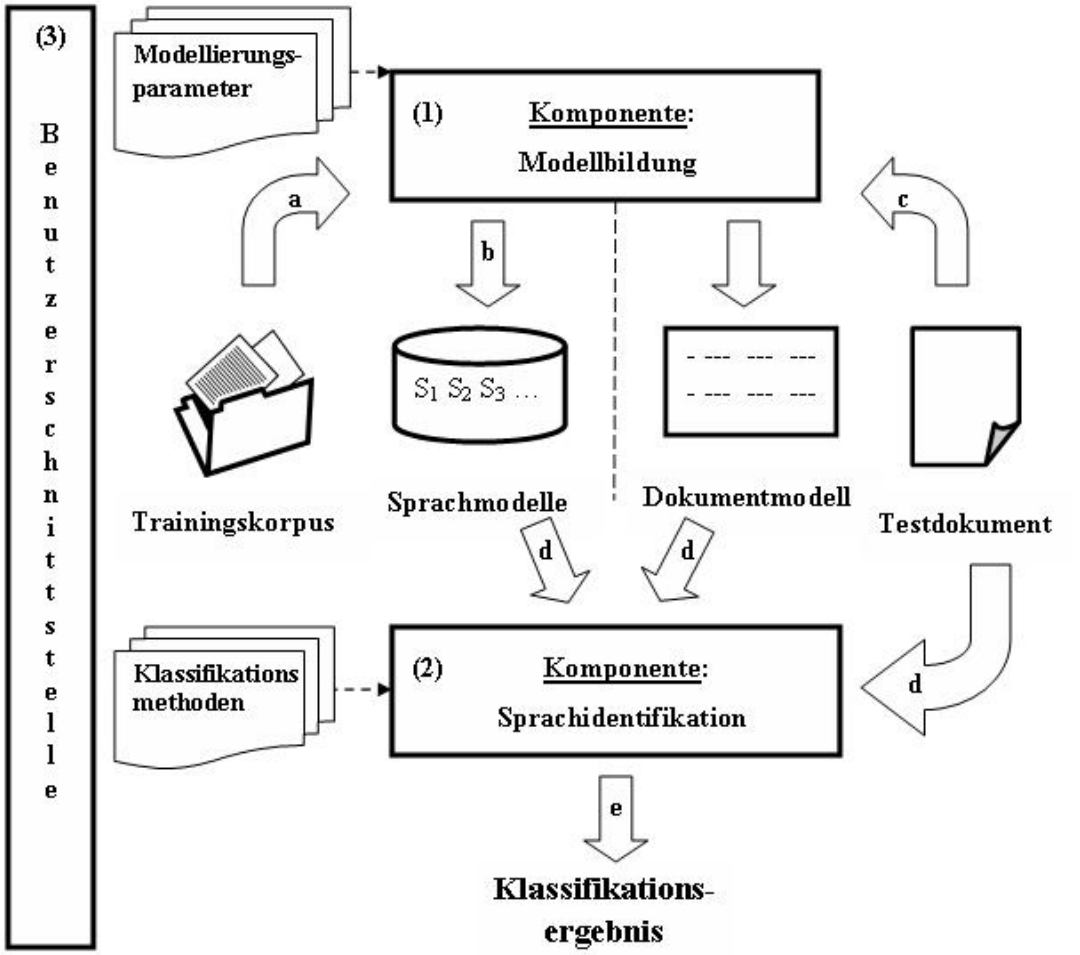


Abbildung 9: Systemarchitektur

Die Systemarchitektur sieht drei Komponenten vor: die Kernkomponente *Modellbildung*, die Kernkomponente *Sprachidentifikation* und die Komponente *Benutzerschnittstelle*. Der Grund für diese Komponentenaufteilung liegt in dem für viele Ansätze typischen Ablauf des Sprachidentifikationsprozesses.

Wie im Kapitel 1 bereits erwähnt, besteht ein typischer Sprachidentifikationsprozess aus zwei Hauptphasen: einer Trainingsphase und einer Erkennungsphase. Diese Phasenaufteilung wird ebenfalls in der oben dargestellten Architektur berücksichtigt. Die Trainingsphase wird hier durch die Kernkomponente Modellbildung, und die Erkennungsphase wird entweder durch die Kernkomponente Sprachidentifikation oder durch beide Kernkomponenten realisiert. Die Komponente Benutzerschnittstelle ermöglicht die Interaktion des Benutzers mit den Kernkomponenten und steuert dabei den Datenverarbeitungsprozess.

Der in dem obigen Schema abgebildete Datenverarbeitungsprozess lässt sich in folgende Schritte gliedern:

a. Der Prozess startet mit der Verarbeitung von Daten aus dem Trainingskorpus durch die Kernkomponente Modellbildung. Als Ergebnis liefert die Komponente eine Menge von erzeugten Sprachmodellen. Für die Erstellung von Sprachmodellen werden zwei unterschiedliche Spracheinheiten verwendet: Tri-Gramme und Wörter. Die Modellbildung erfolgt gemäß den eingestellten Modellierungsparametern.

   Die Auswahl von Trainingsdaten und von Modellierungsparametern wird über die Komponente Benutzerschnittstelle festgelegt.

b. Die erzeugten Sprachmodelle werden gespeichert, um zu einem späteren Zeitpunkt für den Modellvergleich verwendet zu werden.

c. Der Verlauf des nächsten Schrittes hängt davon ab, welche der Klassifikationsmethoden im Sprachidentifikationsprozess eingesetzt wird. Entweder wird das eingegebene Testdokument auf die gleiche Weise wie die Trainingsdokumente durch die Kernkomponente Modellbildung verarbeitet, oder es wird direkt nach dem Schritt b zum Schritt d übergegangen.

d. Das erstellte Testdokumentmodell bzw. das Testdokument wird durch die Komponente Sprachidentifikation mit den zuvor erzeugten Sprachmodellen verglichen.

   Die Auswahl der Klassifikationsmethoden wird vom Benutzer über die Benutzerschnittstelle bestimmt.

e. Als Ergebnis der oben beschriebenen Schritte werden die Sprachen ausgegeben, in denen das Testdokument am wahrscheinlichsten verfasst ist.

9 Kernkomponenten des Systems

In diesem Kapitel werden die Systemkernkomponenten Modellbildung und Sprachidentifikation näher betrachtet. Die Realisierung der Komponenten erfolgt unter der Anwendung unterschiedlicher Sprachmodellierungs- und Klassifikationsverfahren, die in den nachfolgenden Unterkapiteln in Pseudocode dargestellt werden.

9.1 Sprachmodellbildung

Da die Sprachidentifizierung bei LangIdent mit Hilfe von Tri-Gramm und wort-basierten Verfahren erfolgt, werden bei der Sprachmodellbildung die Informationen sowohl zu der Tri-Gramm- als auch zu der Wortverteilung erfasst. Dementsprechend besteht jedes der acht Sprachmodelle aus zwei Listen: einer Tri-Gramm-Liste und einer Wortliste. Als Datenstruktur zum Speichern der Spracheinheiten wurde die Hashtabelle (engl. hashtable) gewählt, da sie einen schnellen Zugriff auf ihre Elemente erlaubt. In den Fällen, wo die Sortierung der Spracheinheiten erwünscht ist, werden die Elemente vorübergehend in einer Baumstruktur oder einem Array gespeichert.
Der genaue Verlauf des Sprachmodellierungsprozesses wird in folgenden Unterkapiteln algorithmisch dargestellt.

9.1.1 Aufbau von Trainingskorpus

Für die Sprachmodellbildung wird ein Trainingskorpus zusammengestellt, der in elektronischer Form vorliegende und bzgl. deren Sprache vorklassifizierte Dokumente enthält.
Bevor der Trainingskorpus aufgebaut wird, soll die Entscheidung bzgl. der Anzahl und Auswahl der Sprachen getroffen werden, die vom System erkannt werden sollen.

Da das zu entwickelnde System jederzeit um weitere Sprachen erweiterbar sein sollte, wurden nur wenige Sprachen in das Trainingsset aufgenommen, um den Zeitaufwand für die Erstellung der Trainings- und Testkorpora sowie für die Evaluierung möglichst gering zu halten.

Bei der Sprachauswahl spielte die Kenntnis der zu identifizierenden Sprachen eine wichtige Rolle, weil dies für die Entwickler den Prozess der Identifikation, Analyse und Eliminierung von Fehlern bei der Zuordnung von Sprachen erleichtert. Zuerst wurde die Sprachauswahl auf sechs Sprachen begrenzt: Deutsch, Englisch, Spanisch, Italienisch, Französisch und Russisch, die von den Entwicklern in einem bestimmten Maß beherrscht werden.

Im Laufe der Systemevaluierung wurden zwei weitere Sprachen hinzugefügt: Tschechisch und Ukrainisch. Dies sollte Aufschluss darüber geben, wie sich das Hinzufügen von weiteren Sprachen auf die durchschnittliche Systemperformanz auswirken würde. Insbesondere war es interessant zu erfahren, wie sich die Erkennungsquoten des Russischen als der einzigen kyrillischen Sprache im Trainingskorpus ändern, falls eine weitere Sprache des kyrillischen Alphabets hinzugefügt wird.

Für den Aufbau des Trainingskorpus wurden Online Zeitungen benutzt, wobei Artikel aus verschiedenen Themenbereichen nach Zufallsprinzip gewählt wurden: Wirtschaft, Politik, Gesellschaft, Wissenschaft, Kultur, Gesundheit etc. Die einzelnen Artikel in jeder Sprache wurden zu einem Text zusammengefügt, so dass im Endeffekt der Trainingskorpus für jede unterstützte Sprache eine große *.txt* - Datei enthielt. Insgesamt wurde pro Sprache ca. 200 KByte in Unicode Big Endian kodierten Text benutzt.

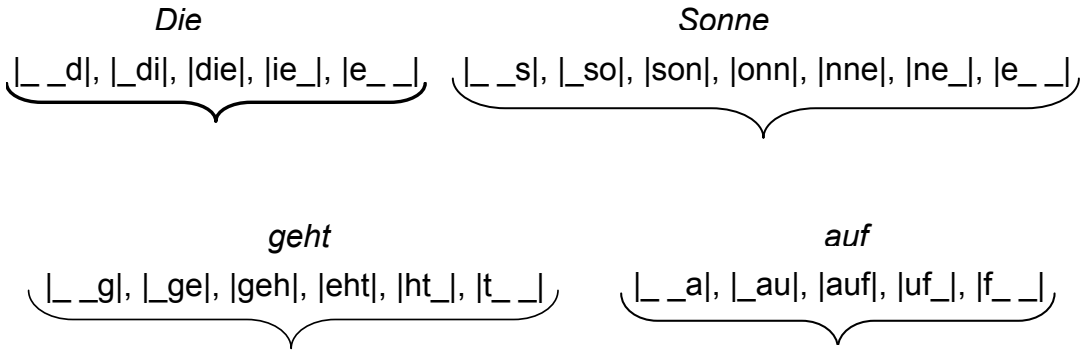
Für den Korpusaufbau wurden folgende Quellen verwendet:

- Deutsch: *Die Tageszeitung* (http://www.taz.de/pt/.nf/home), *Netzeitung.de* (http://www. netzeitung.de), 205 KB
- Englisch*: BBC News* (http://news.bbc.co.uk/), 203 KB
- Französisch: *LeFfigaro.fr* (http://www.lefigaro.fr/), 197 KB
- Italienisch: *La Stampa web* (http://www.lastampa.it), *L'Unione Sarda.it* (http://www.unionesarda.it), 204 KB
- Spanisch: *El Mundo.es* (http://www.elmundo.es/), *El Tiempo.com* (http://eltiempo.terra.com.co/), 207 KB

- Russisch: *Газета.ru* (http://www.gazeta.ru), 203 KB
- Ukranisch: *ДзеркалоТижня* (http://www.zn.kiev.ua), 206 KB
- Tschechisch: *Lidovsky Centrum.cz* (http://lidovky. centrum.cz), 203 KB

9.1.2 Tri-Gramm basierte Sprachmodellbildung

Im ersten Schritt der Modellerstellung wird der eingegebene Trainingstext normalisiert, d. h. alle unerwünschten Zeichen bzw. alle für die Sprachmodellierung irrelevanten Informationsträger wie Zahlen und Sonderzeichen werden eliminiert. Der für die Modellbildung verwendete normalisierte Text soll im Endeffekt nur Buchstaben, Leerzeichen und Apostrophe enthalten. Darüber hinaus werden alle Großbuchstaben in Kleinbuchstaben umgewandelt, um die Anzahl der unterschiedlichen Tri-Gramme zu reduzieren und damit verbundene Verzerrungen der absoluten Tri-Gramm-Häufigkeiten zu vermeiden.
Im nächsten Schritt werden zusätzliche Leerzeichen zwischen den Wörtern eingefügt, so dass zwischen zwei benachbarten Wörtern immer zwei Leerzeichen liegen. Dadurch wird vermieden, dass ein Tri-Gramm Zeichen unterschiedlicher Wörter enthält. Zum einen wird auf diese Weise die Anzahl unterschiedlicher Tri-Gramme reduziert, zum anderen werden durch die auf diese Weise ermittelten Wortanfänge und –enden wichtige Informationen erhalten. Die Tatsache, dass beispielsweise der Buchstabe „n“ im Deutschen sehr häufig am Wortende vorkommt, scheint den Autoren für die Sprachidentifizierung wichtiger zu sein, als dass das darauf folgende Wort mit dem Buchstaben „a“ anfangen kann.
Der normalisierte Text wird Zeichen für Zeichen durchgegangen, und jedes Textfenster der Länge n=3 als ein Tri-Gramm erfasst. Dieser Vorgang kann an einem Beispiel veranschaulicht werden:
Nach der Normalisierung und Einsetzung von zusätzlichen Leerzeichen sieht der Satz „Die Sonne geht auf“ wie folgt aus: _ _die_ _sonne_ _ geht_ _ auf_ _.
Nach der Zerteilung des Texts in Tri-Gramme ergibt sich folgendes Bild:



Die Textnormalisierung kann mittels Pseudocode wie folgt dargestellt werden:

```
INPUT: Dokument: Text

FOR alle Zeichen aus dem Dokument DO

     IF Zeichen ein Großbuchstabe THEN
        Umwandele Großbuchstaben in einen Kleinbuchstaben

     IF Zeichen weder ein Buchstabe noch ein Apostroph THEN
        Ersetze das Zeichen durch ein Leerzeichen

     IF Zeichen ein Leerzeichen AND weder das Zeichen vor
         noch das Zeichen nach dem Zeichen ein Leerzeichen
         THEN
        füge ein Leerzeichen ein

END
```

Abbildung 10: Textnormalisierung (Pseudocode)

Obwohl für die Sprachidentifikation ausschließlich Tri-Gramme eingesetzt werden, werden bei der Modellbildung drei N-Gramm Listen erstellt, die alle in dem Trainingsdokument gefundenen Uni-Gramme, Bi-Gramme und Tri-Gramme enthalten.

Die Uni-Gramm-Liste wird benötigt, um die Größe des Alphabets zu ermitteln, die später für die Berechnung von Übergangswahrscheinlichkeiten der Tri-Gramme verwendet wird. Das Alphabet ist in diesem Fall als Menge unterschiedlicher Zeichen, die in der gegebenen Sprache vorkommen können,

zu verstehen. Ist das Trainingsdokument lang genug, kann davon ausgegangen werden, dass alle Zeichen des Alphabets darin mindestens einmal vorkommen.
Die Trainingsdokumente sollten nach Möglichkeit manuell bereinigt werden, um sicherzustellen, dass sie keine Zeichen aus einem anderen Alphabet enthalten. Beispielsweise können lateinische Buchstaben oder römische Zahlen in einem russischen Text zu Verfälschung der Übergangswahrscheinlichkeiten der Tri-Gramme führen.
Da lediglich die Länge der Uni-Gramm-Liste von Bedeutung ist, kann sie anschließend wieder gelöscht werden.
Die Erstellung der Bi-Gramm-Liste ist notwendig, um die Übergangswahrscheinlichkeiten der in der Tri-Gramm-Liste enthaltenen Tri-Gramme zu berechnen (vgl. Punkt 3.3.2, S. 40). Sie muss mitgespeichert werden, damit in der Erkennungsphase die Übergangswahrscheinlichkeiten der Tri-Gramme, die nicht im Sprachmodell gefunden werden, ermittelt werden können.
Während der Listenerstellung werden absolute Häufigkeiten aller in den Trainingstexten gefundenen Bi- und Tri-Gramme berechnet. Der unten abgebildete Algorithmus gibt den Vorgang wieder. Aus Gründen der Übersichtlichkeit enthält er drei Schleifen, die in der Implementierung zu einer zusammengefasst werden.

```
INPUT: normalisiertes Dokument: Text
DEFINE UnigramTable, BigramTable, TrigramTable: Hashtable
FOR alle Zeichen aus dem normalisierten Dokument DO
     IF Zeichen nicht Leerzeichen THEN
          IF Zeichen nicht in UnigramTable vorhanden THEN
             Speichere Zeichen in UnigramTable,
          setze Häufigkeit für Zeichen gleich 1
          ELSE inkrementiere Häufigkeit für Zeichen um 1
END
```

```
FOR alle Teilstrings der Länge 2 aus dem normalisierten
    Dokument DO
     IF Teilstring nicht in BigramTable vorhanden THEN
        Speichere Teilstring in BigramTable,
     setze Häufigkeit für Teilstring auf 1
     ELSE inkrementiere Häufigkeit für Teilstring um 1
END
FOR alle Teilstrings der Länge 3 aus dem normalisierten
    Dokument DO
     IF Teilstring nicht in TrigramTable vorhanden THEN
        Speichere Teilstring in TrigramTable,
setze Häufigkeit für Teilstring auf 1
END
```

Abbildung 11: Erstellen von Hashtabellen (Pseudocode)

Im nächsten Schritt werden die Tri-Gramme absteigend nach ihrer absoluten Häufigkeit sortiert.

Die im vorherigen Schritt ausgerechneten absoluten Häufigkeiten der aus den Trainingsdaten gewonnenen Tri-Gramme werden für die Berechnung der relativen Häufigkeiten und der inversen Dokumenthäufigkeiten benutzt.

Die relativen Häufigkeiten werden ermittelt, indem absolute Häufigkeit $H_d(x)$ des jeweiligen Tri-Gramms x durch die Anzahl N_d aller im Trainingstext d der jeweiligen Sprache vorkommenden Tri-Gramme dividiert wird:

$$RH_d(x) = \frac{H_d(x)}{N_d}$$

Wird beispielsweise das Tri-Gramm *„der“* in einem deutschen Trainingsdokument 840 Mal gefunden (840 stellt den absoluten Häufigkeitswert dar) und ist die

Gesamtzahl aller Tri-Gramme in demselben Text gleich 117318, beträgt die relative Häufigkeit in diesem Fall: 840 /117318 ≈ 0,00716.
Mit Hilfe der inversen Dokumenthäufigkeit wird die Nützlichkeit eines N-Gramms zur Unterscheidung von Sprachen bestimmt. Dieses Maß bevorzugt N-Gramme, die in einer Sprache eher häufig und in den anderen Sprachen eher selten vorkommen. Die inverse Dokumenthäufigkeit wird berechnet, indem die Häufigkeit $H_d(x)$ eines Tri-Gramms x im Dokument d durch die Anzahl n der Sprachmodelle, in denen x vorkommt, dividiert wird:

$$idf_d(x) = \frac{H_d(x)}{n}$$

Wenn beispielsweise ein Tri-Gramm, das in einem deutschen Trainingsdokument 20 Mal vorkommt, in den Tri-Gramm-Listen vier weiterer Sprachemodelle enthalten ist, ist seine inverse Häufigkeit gleich 4: 20 /5 = 4. Würde dieses Tri-Gramm ausschließlich im deutschen Sprachmodell vorhanden sein, dann würde die inverse Häufigkeit auf 20 steigen: 20/1 = 20. In dem Sprachidentifizierungssystem LangIdent werden dynamische Sprachmodelle verwendet. Jedes Mal, wenn ein Sprachmodell hinzugefügt oder gelöscht wird, werden die inversen Häufigkeiten aller Sprachmodelle neu berechnet.
Die Berechnung von Übergangswahrscheinlichkeiten erfolgt nach der Formel von T. Dunning, die im Kapitel 3 beschrieben wurde (s. Punkt 3.3.2, S. 41):

$$\hat{p}(w_{k+1} \mid w_1 \ldots w_k \mid A) = \frac{T(w_1 \ldots w_{k+1}, T_A) + 1}{T(w_1 \ldots w_k, T_A) + m}$$

Im System ist es vorgesehen, dass dem Benutzer bei der Erstellung neuer Sprachmodelle mehrere Optionen zur Auswahl stehen. Er kann festlegen, dass alle im Trainingsdokument gefundenen Tri-Gramme in das Sprachmodell aufgenommen werden, es können aber auch die maximale Länge der Tri-Gramm-Liste oder die minimale absolute Häufigkeit der in der Liste enthaltenen Tri-Gramme angegeben werden. Die Tri-Gramm-Liste muss dann entsprechend angepasst werden.

Die Voreinstellung ist, dass die erstellten Tri-Gramm-Listen auf 1500 häufigste Tri-Gramme reduziert werden, d.h. nur die 1500 häufigsten Tri-Gramme werden in dem Sprachmodell gespeichert.

Tri-Gramm	Absolute Häufigkeit	Relative Häufigkeit	Übergangswahr-Scheinlichkeit	Inverse Dokument-häufigkeit
|n |	3606	0,03075	0,99148	721,2
|en |	2469	0,02106	0,71802	617,25
|e |	2250	0,01919	0,98642	450,0
| d|	2231	0,01903	0,13558	446,2
|r |	1995	0,01701	0,98471	399,0
|er |	1499	0,01278	0,46069	299,8
|t |	1486	0,01267	0,97958	297,2
| s|	1277	0,01089	0,07763	255,4
|s |	1243	0,0106	0,97569	248,6
| a|	1056	0,00901	0,0642	211,2
| de|	1021	0,00871	0,45161	204,2
| e|	987	0,00842	0,06001	197,4
|der|	840	0,00716	0,44973	168,0
| w|	818	0,00698	0,04975	163,6
|sch|	790	0,00674	0,94958	263,33333
…	…	…	…	…

Tabelle 10: Die 15 häufigsten Tri-Gramme aus dem deutschen Sprachmodell

Ein Ausschnitt (15 Tri-Gramme) aus der auf die oben beschriebene Art und Weise erstellten Tri-Gramm-Liste wird in der oben dargestellten Tabelle präsentiert. Diese Liste ist ein Bestandteil des bei LangIdent gespeicherten deutschen Sprachmodells, das anhand von 205 KByte Trainingsdaten erzeugt wurde

Aus der Tabelle lässt sich zum Beispiel erkennen, dass *n* die häufigste Endung im Deutschen ist, gefolgt von *en* und *e*; dass die Wörter am häufigsten mit *d* anfangen; dass das Tri-Gramm *der* in den Tri-Gramm-Listen vier weiterer Sprachmodelle enthalten ist; dass die Wahrscheinlichkeit dafür, dass dem Bi-Gramm *sc* im Deutschen ein *h* folgt, bei ca. 95% liegt u. v. m.

9.1.3 Wortbasierte Sprachmodellbildung

Bei der wortbasierten Modellbildung wird der normalisierte Text in einzelne Wörter bzw. Wortformen, die sogenannten Tokens segmentiert. Danach werden für jedes Wort absolute und relative Häufigkeiten berechnet. Basierend auf ihren absoluten Häufigkeiten werden die Wörter absteigend sortiert. Die kumulativen Häufigkeiten können jetzt ermittelt werden, indem für jedes Wort die relativen Häufigkeiten der vor ihm in der Wortliste platzierten Wörter einschließlich seiner eigenen summiert werden. So ist die kumulative Häufigkeit des ersten Wortes gleich seiner relativen Häufigkeit, des zweiten – der Summe der relativen Häufigkeiten der beiden ersten Wörter usw. Die kumulative Häufigkeit ist eine dynamische Größe, was bedeutet, dass sie jedes Mal nach Bearbeitung der Wortlisten neu berechnet wird.

Je nachdem, welche Einstellungen bei der Modellerstellung gewählt wurden, werden entweder alle Wörter aus der Liste in das Sprachmodell aufgenommen, oder nur die Wörter, die vom Benutzer festgelegte maximale Länge nicht überschreiten, oder die N häufigsten Wörter, oder die häufigsten Wörter, deren kumulative Häufigkeit einen bestimmten Prozentsatz erreicht. In den von den Autoren erstellten Sprachmodellen enthalten die Wortlisten die häufigsten Wörter, die 40% des standardisierten Texts ausmachen.

Wort	Absolute Häufigkeit	Relative Häufigkeit	Kumulative Häufigkeit	Inverse Dokument-Häufigkeit
die	563	0,03953	0,03953	563,0
der	552	0,03876	0,07829	552,0
und	319	0,0224	0,10069	319,0
in	239	0,01678	0,11747	79,66667
den	184	0,01292	0,13039	184,0
zu	166	0,01165	0,14204	166,0
das	160	0,01123	0,15327	160,0
von	152	0,01067	0,16394	152,0
sich	128	0,00899	0,17293	128,0
für	121	0,0085	0,18143	121,0
nicht	120	0,00843	0,18986	120,0
mit	96	0,00674	0,1966	96,0
auf	94	0,0066	0,2032	94,0
ist	94	0,0066	0,2098	94,0
im	92	0,00646	0,21626	92,0
…	…	…	…	…

Tabelle 11: Die 15 häufigsten Wörter aus dem deutschen Sprachmodell

Die Tabelle zeigt die ersten 15 Wörter des deutschen Sprachmodells. Es kommt nicht überraschend, dass die ersten Plätze durch Artikel, Präpositionen und Konjunktionen belegt werden, die sehr hohe relative Häufigkeiten aufweisen. Die ersten 12 Wörter der Liste decken fast 20% des standardisierten Texts. Diese Verteilung ist typisch für die meisten westeuropäischen Sprachen.

Bei stark flektierenden Sprachen, wie z.B. Russisch oder Ukrainisch, haben die häufigsten Wörter wesentlich niedrigere relative Häufigkeiten, so dass die Wortlisten insgesamt länger sind (vgl. 23 Wörter im spanischen Sprachmodell vs. 234 Wörter im ukrainischen Modell).

Eine weitere Besonderheit der Wortlisten von Ukrainisch und Russisch besteht darin, dass sie viele dokumentspezifische Wörter enthalten, oft mehrere Formen desselben Wortes. Die Substantive erscheinen ziemlich früh in der Wortliste. So kommt im ukrainischen Sprachmodell bereits auf der Position 26 das Substantiv *рішення* (dt. Entscheidung, Lösung), im Russischen ebenfalls auf der Position 26 das Wort *время* (dt. Zeit). In der deutschen Wortliste erscheint das erste Substantiv dagegen erst auf Platz 73. Eine mögliche Erklärung für dieses Phänomen könnte in der Tatsache liegen, dass die russischen und ukrainischen Funktionswörter verhältnismäßig niedrige relative Häufigkeiten aufweisen, die mit denen der dokumentspezifischen Wörter vergleichbar sind.

Daraus lässt sich schließen, dass die gewählte Vorgehensweise bei der Erstellung der Wortlisten für die stark flektierenden Sprachen nicht zu den gewünschten Ergebnissen führt. Eine Alternative zur Angabe des Cut-Off-Wertes könnte die Erstellung einer längeren Wortliste von 500-600 Wörtern darstellen, die anschließend manuell bearbeitet wird. Die dokumentspezifischen Wörter werden dabei aus der Wortliste entfernt, bis die kumulative Häufigkeit den gleichen Wert wie bei den anderen Sprachmodellen erreicht.
Alle von den Autoren erstellten Sprachmodelle wurden auf die gleiche Weise erstellt, und zwar mit der Angabe der kumulativen Häufigkeit von 40%, um die Ergebnisse besser vergleichen zu können.

9.2 Sprachidentifikation monolingualer Dokumente

Für die Sprachidentifikation monolingualer Texte werden bei LangIdent drei verschiedene Klassifikationsmethoden eingesetzt: das Ad Hoc Ranking Verfahren (out of place Methode), das Vektorraum Modell und die Bayes'sche Entscheidungsregel. Außerdem kann die Sprache anhand des wortbasierten Vergleichs bestimmt werden. Der Benutzer kann selbst entscheiden, welche der oben genannten Methoden zur Sprachidentifikation verwendet werden soll. Es ist ebenfalls möglich, alle vier Methoden einzusetzen. In diesem Fall werden vom System vier Ergebnisse zurückgeliefert.
Werden das Vektorraum Modell oder das Ad Hoc Ranking Verfahren für die Spracherkennung eingesetzt, muss zuerst das Modell des eingegebenen Testdokuments erstellt werden. Entscheidend für die Zusammensetzung des Modells ist dabei die gewählte Klassifikationsmethode. Für das Ad Hoc Ranking Verfahren wird lediglich eine Tri-Gramm-Liste erstellt. Entscheidet sich der Benutzer für das Vektorraum Modell, muss für das Dokumentmodell außerdem eine Wortliste gebildet werden. Das gleiche gilt, falls alle vier Methoden gewählt wurden.
Der Prozess der Modellbildung verläuft ähnlich wie bei der Erstellung der Sprachmodelle. Die wesentlichen Unterschiede bestehen darin, dass erstens die Uni-Gramm- und Bi-Gramm-Listen nicht erstellt werden müssen, und zweitens

die Übergangswahrscheinlichkeiten und die inversen Dokumenthäufigkeiten nicht berechnet werden.
Nachdem das Dokumentmodell erstellt wurde, kann es mit den im System gespeicherten Sprachmodellen verglichen werden, um das ihm ähnlichste Modell auszuwählen. Als Ähnlichkeitsmaß dient bei der Ad hoc Ranking Methode die Distanz zwischen dem Dokumentmodell und dem Sprachmodell, bei dem Vektorraum Modell ist dies die Kosinusdistanz zwischen dem Dokumentmodell-Vektor und dem Sprachmodell-Vektor.
Beim Einsatz der Bayes'schen Entscheidungsregel oder des wortbasierten Verfahrens ist die Erstellung des Dokumentmodells nicht erforderlich. Das eingegebene Dokument muss lediglich normalisiert werden. Der Normalisierungsschritt ist mit dem im Punkt 9.1.2 beschriebenen identisch.
Im Weiteren werden die einzelnen Verfahren kurz dargestellt. Die theoretischen Grundlagen können im Kapitel 3 nachgelesen werden.

Ad Hoc Ranking Verfahren

Das Ad Hoc Ranking Verfahren, das in LangIdent implementiert wird, beruht auf dem im Kapitel 3 beschriebenen Algorithmus von W. B. Cavnar & J. M. Trenkle (s. Punkt 3.1, S. 32).
Für die Berechnung der Distanz zwischen einem Dokument- und einem Sprachmodell werden die Tri-Gramm-Listen beider Modelle zunächst entsprechend der absoluten Häufigkeiten der Tri-Gramme in absteigender Reihenfolge sortiert. Ausgehend von der Reihenfolge erhält jedes Tri-Gramm eine Position in der jeweiligen Liste.
Anschließend wird jedes Tri-Gramm aus dem Dokumentmodell im Sprachmodell gesucht. Im Falle, dass es gefunden wird, wird die Distanz um den absoluten Betrag der Differenz zwischen der Position des Tri-Gramms im Sprachmodell und seiner Position im Dokumentmodell erhöht. Ist das Tri-Gramm nicht in dem Sprachmodell enthalten, wird zu der Distanz ein maximaler Out-of-place-Wert addiert. Die Arbeit von Cavnar & Trenkle enthält keine Angaben zu der Höhe des Out-of-place-Wertes. Es liegt nah, ihn auf N+1 zu setzen, wobei N die Anzahl der

Tri-Gramme in der Tri-Gramm-Liste des Sprachmodells darstellt. Diese Herangehensweise eignet sich aber nur dann, wenn die Tri-Gramm-Listen aller Sprachmodelle die gleiche Länge haben. Dies ist jedoch nur dann der Fall, wenn bei der Sprachmodellbildung die Option *die N häufigsten Tri-Gramme* gewählt wurde.

Cowie et al. haben bei dem von ihnen implementierten System den Wert von 10000 benutzt. Sie berichten, dass die Fehlerquoten dabei im Vergleich mit der oben beschriebenen Methode um den Faktor 8 reduziert werden konnten (vgl. COWIE, J. et al. 1999: 5).

Bei LangIdent wird der Out-of-place-Wert ebenfalls auf 10000 gesetzt.

```
INPUT: Dokumentmodell, Sprachmodell

DEFINE Distanz := 0

FOR jedes Tri-Gramm aus Dokumentmodell DO

     IF Tri-Gramm im Sprachmodell vorhanden THEN

          Distanz := Distanz + Betrag der Differenz
          zwischen der Position des Tri-Gramms im
          Dokumentmodell und seiner Position im
          Sprachmodell

     ELSE

          Distanz := Distanz + 10000

END
```

Abbildung 12: Berechnung der Distanz zwischen dem Dokumentmodell und einem der Sprachmodelle (Pseudocode)

Nachdem die Distanzen aller Sprachmodelle zu dem Dokumentmodell berechnet wurden, wird das Sprachmodell mit der kleinsten Distanz ausgewählt. Die mit ihm assoziierte Sprache wird von dem System als die Dokumentsprache zurückgegeben.

Vectorraum Modell

Bei dem Vektorraum Modell wird nach der Sprache gesucht, deren Sprachmodellvektor dem Dokumentvektor am ähnlichsten ist. Als Ähnlichkeitsmaß wird dabei die so genannte Kosinusdistanz verwendet (vgl. Punkt 3.2, S. 34).
In dem von den Autoren implementierten System werden als Features oder Vektorenwerte bei den Dokumentmodellen die absoluten Häufigkeiten der Tri-Gramme und der Wörter eingesetzt, bei den Sprachmodellvektoren sind das die inversen Dokumenthäufigkeiten. Die Tri-Gramme und Wörter stellen dabei nicht etwa verschiedene Vektoren dar, sondern werden zu einem Vektor zusammengefasst.
Die Wörter der Sprachmodellvektoren erhalten eine höhere Gewichtung als Tri-Gramme. Die optimale Gewichtung wurde empirisch bestimmt. Das System wurde an einem Subset des Trainingskorpus mit ca. 100 Dokumenten pro Sprache getestet. Die besten Ergebnisse wurden bei der Multiplizierung der inversen Dokumenthäufigkeiten der Wörter mit dem Faktor 6 erzielt.
Um die Überbewertung der Tri-Gramme, die auch in der Wortliste vorkommen, zu vermeiden, werden sie bei der Berechnung der Kosinusdistanz nicht berücksichtigt.
Der Pseudocode unten demonstriert die Berechnung der Kosinusdistanz zwischen dem Dokumentvektor und einem der Sprachmodellvektoren:

```
INPUT: Dokumentmodell, Sprachmodell

DEFINE Skalarprodukt := 0, Norm1 =: 0, Norm2 =: 0

FOR jedes Tri-Gramm aus Dokumentmodell DO

     IF Tri-Gramm in der Tri-Gramm-Liste des
Sprachmodells jedoch nicht in der Wortliste vorhanden
THEN
```

```
        Skalarprodukt := Skalarprodukt +  absolute
        Häufigkeit des Tri-Gramms im Dokumentmodell *
        inverse Häufigkeit des Tri-Gramms im
        Sprachmodell
        Norm1 := Norm1 + (absolute Häufigkeit des Tri-
        Gramms im Dokumentmodell)²
        Norm2 := Norm2 +(inverse Häufigkeit des Tri-
        Gramms im Sprachmodell)²
END
FOR jedes Wort aus Dokumentmodell DO
    IF Wort im Sprachmodell vorhanden THEN
        Skalarprodukt := Skalarprodukt + 6 * absolute
        Häufigkeit des Wortes im Dokumentmodell *
        inverse Häufigkeit des Wortes im Sprachmodell
        Norm1 := Norm1 + (absolute Häufigkeit des
        Wortes im Dokumentmodell)²
        Norm2 := Norm2 + (inverse Häufigkeit Wort des
        Wortes im Sprachmodell)²
END
RETURN Skalarprodukt / Wurzel((Norm1)*(Norm2))
```

Abbildung 13: Berechnung der Kosinusdistanz zwischen dem Dokumentmodellvektor und einem der Sprachmodellvektoren

Als Dokumentsprache wird die Sprache zurückgeliefert, deren Sprachmodellvektor die größte Kosinusdistanz und somit den kleinsten Winkel zu dem Dokumentvektor aufweist.

Im Großen und Ganzen ist die Umsetzung des Vektorraum Modells der in der Arbeit von Prager beschriebenen Implementierung sehr ähnlich (vgl. PRAGER, J.M. 1999).

Bayes'sche Entscheidungsregel

Bei der Verwendung der Bayes'schen Entscheidungsregel für die Sprachidentifizierung wird für jede Sprache die Wahrscheinlichkeit dafür berechnet, dass das Testdokument in dieser Sprache verfasst ist. Die Berechnung der Wahrscheinlichkeit erfolgt nach der Formel von T. Dunning (s. Punkt 3.3.2, S. 39):

$$p(S \mid A) = \prod_{w_1 \dots w_{k+1} \in S} p(w_{k+1} \mid w_1 \dots w_k \mid A) \cdot T(w_1 \dots w_{k+1}, S)$$

Die Erstellung eines Dokumentmodells ist nicht erforderlich. Der normalisierte Text wird in Tri-Gramme geteilt. Das Produkt der Übergangswahrscheinlichkeiten aller im Testdokument gefundenen Tri-Gramme ergibt die Wahrscheinlichkeit dafür, dass das Testdokument in der gegebenen Sprache verfasst ist. Um die numerischen Fehler zu vermeiden, kann stattdessen die Summe der Logarithmen der Übergangswahrscheinlichkeiten benutzt werden (s. Punkt 3.3.2, S. 40):

$$\log\, p(S \mid A) = \sum_{w_1 \dots w_{k+1} \in S} T(w_1 \dots w_{k+1}, S) \log\, p(w_{k+1} \mid w_1 \dots w_k \mid A)$$

Die Übergangswahrscheinlichkeiten der in dem Sprachmodell enthaltenen Tri-Gramme werden bereits bei der Sprachmodellerstellung berechnet, können also direkt in die Formel eingesetzt werden.
Wird ein Tri-Gramm in dem Sprachmodell nicht gefunden, kann seine Übergangswahrscheinlichkeit nach der Formel von T.Dunning berechnet werden (s. Punkt 3.3.2, S. 41):

$$\hat{p}(w_{k+1} \mid w_1 \dots w_k \mid A) = \frac{T(w_1 \dots w_{k+1}, T_A) + 1}{T(w_1 \dots w_k, T_A) + m},$$

wobei der Wert $T(w_1 \ldots w_{k+1}, T_A)$ gleich Null ist, $T(w_1 \ldots w_k, T_A)$ die absolute Häufigkeit des entsprechenden Bi-Gramms darstellt, falls es in der Bi-Gramm-Liste des Sprachmodells enthalten ist, andernfalls ist es gleich Null. Die Variable m steht für die Alphabetgröße der jeweiligen Sprache.
Der oben beschriebene Algorithmus wird nachstehend in Pseudocode dargestellt:

```
INPUT: normalisierter Text, Sprachmodell
DEFINE Wahrscheinlichkeit := 0, Häufigkeit:=0
FOR jedes Tri-Gramm aus dem normalisierten Text DO
     IF  Tri-Gramm im Sprachmodell enthalten THEN
     Wahrscheinlichkeit := Wahrscheinlichkeit +
     LOG(Übergangswahrscheinlichkeit des Tri-Gramms)
          ELSE
               IF   Bi-Gramm im Sprachmodell enthalten
     THEN
          Häufigkeit := absolute Häufigkeit des Bi-
     Gramms
     ELSE Häufigkeit := 0
     Wahrscheinlichkeit := 1/(Häufigkeit + die Größe des
     Alphabets des Sprachmodells)
END
RETURN Wahrscheinlichkeit
```

Abbildung 14: Bestimmung der Wahrscheinlichkeit für ein Sprachmodell, dass das Testdokument in der mit ihm assoziierten Sprache verfasst ist (Pseudocode)

Das Sprachmodell mit der höchsten Wahrscheinlichkeit bzw. mit dem kleinsten Logarithmus repräsentiert die Dokumentsprache.

Die wortbasierte Methode

Bei dieser Methode werden ebenfalls keine Dokumentmodelle erstellt, sondern die Testdokumente werden direkt mit den Sprachmodellen verglichen.
Das eingegebene Testdokument wird normalisiert, in einzelne Wörter zerlegt und anschließend Wort für Wort iteriert. In jedem Iterationsschritt wird untersucht, in welchen Sprachmodellen das jeweilige Wort vorkommt.
In der ersten Implementierung der wortbasierten Methode wurden lediglich die relativen Häufigkeiten der gefunden Wörter summiert, und anschließend die Sprache mit der höchsten Summe für die Dokumentsprache erklärt. Dies führte jedoch zu Problemen bei der Identifikation eng verwandter Sprachen, wie z. B. Französisch, Italienisch und Spanisch. Die hoch frequenten Wörter, die in allen drei Sprachen vorkommen, weisen im Spanischen eine wesentlich höhere relative Häufigkeit auf, so dass viele italienische und französische Texte fälschlicherweise als spanisch identifiziert wurden.
Nach der Analyse der Ergebnisse wurde die wortbasierte Methode modifiziert.
In der aktuellen Version wird für jede Sprache zusätzlich ein Zähler geführt. Wird ein Wort aus dem Testdokument in einem der Sprachmodelle gefunden, wird der Zähler für dieses Sprachmodell inkrementiert. Es wird anschließend das Sprachmodell mit dem höchsten Zählerstand gewählt.
Falls in mehreren Sprachmodellen die gleiche Anzahl Wörter aus dem Testdokument gefunden wird, werden die Summen der relativen Häufigkeiten der gefundenen Wörter bei den Sprachen mit dem höchsten Zählerstand miteinander verglichen. Das Modell mit dem höchsten Summenwert wird als Dokumentsprache bestimmt.

```
INPUT: Dokument, Sprachmodell[Anzahl der Sprachmodelle];

DEFINE Zähler[Anzahl der Sprachmodelle],
     kumulative Häufigkeit[Anzahl der Sprachmodelle],
     maxZähler, maxSumme;
```

```
FOR i=0; i< Anzahl der Sprachmodelle; i++ DO
     FOR jedes Wort aus Dokument DO
          IF Wort im Sprachmodell[i] vorhanden THEN
               Zähler[i] := Zähler[i] +1;
               kumulative Häufigkeit[i] := kumulative
               Häufigkeit[i] + relative Häufigkeit des
               Wortes im Sprachmodell[i];
     END
END
maxZähler = findMaxInteger(Zähler);
IF Zähler[maxZähler] ist gleich 0 THEN
     RETURN -1 (keine Entscheidung möglich)
ELSE
     maxSumme := maxZähler;
     FOR i=0; i<Anzahl der Sprachmodelle; i++ DO
          IF Zähler[i] gleich Zähler[maxZähler] und i
     nicht gleich maxZähler THEN
          IF kumulative Häufigkeit[i]>kumulative
          Häufigkeit[maxZähler]THEN
               maxSumme := i;
     END
     RETURN Sprachmodell[maxSumme]
```

Abbildung 15: Bestimmung der wahrscheinlichsten Sprache des Dokuments mit Hilfe der wortbasierten Methode (Pseudocode)

Die Methode findMaxInteger (int []) ermittelt die Position des größten Elements in einem Array mit ganzen Zahlen. Sie wird hier nicht näher erläutert.

9.3 Sprachidentifikation multilingualer Dokumente

Bei der Analyse der in mehreren Sprachen verfassten Dokumente werden an das Sprachidentifizierungssystem folgende Anforderungen gestellt: erstens müssen alle im Dokument enthaltenen Sprachen korrekt identifiziert werden, zweitens soll die Position, an der ein Sprachwechsel stattfindet, möglichst genau bestimmt werden.

Die Grundidee für die Bestimmung mehrerer Sprachen in einem Dokument wurde der Arbeit von M. J. Martino & R. C. Paulsen entnommen (MARTINO, M. J. & PAULSEN, R. C. 2001). Es wird ein Intervall definiert, das die Größe des Fensters (in Wörtern) festlegt, das immer um ein Wort nach rechts verschoben wird, bis das Dokumentende erreicht wird. An jeder Position wird die Sprache des vom Fenster eingegrenzten Textabschnitts identifiziert.

Stimmt die Sprache des aktuellen Abschnitts mit der Sprache des vorherigen Abschnitts nicht überein, wird im vorliegenden Textabschnitt ein Sprachwechsel angenommen. Die Position des Sprachwechsels kann unter Umständen genauer bestimmt werden, wenn die Satzzeichen und Zeilenumbrüche berücksichtigt werden.

Der Spracherkennung liegt bei Martino der wortbasierte Ansatz zu Grunde. Eine ausführliche Beschreibung des Algorithmus kann im Kapitel 4 nachgelesen werden.

Die Vorteile der oben beschriebenen Methode liegen in der einfachen Implementierung und geringem Speicherbedarf für die Sprachmodelle, da nur einige wenige Wörter pro Sprache gespeichert werden sollen.

Allerdings weist die Methode einige Schwachstellen auf. Nachfolgend werden einige Nachteile angeführt, die dazu beigetragen haben, dass die Autoren sich für den Tri-Gramm basierten Ansatz entschieden haben.

1) Wortlisten verwandter Sprachen enthalten oft gleich lautende Wörter. Wird in dem Fall, wenn ein Wort aus dem Testdokument in einem Sprachmodell gefunden wird, der Zähler für das entsprechende Modell um eins inkrementiert, kann es vorkommen, dass mehrere Zähler den gleichen Stand haben. Eine Entscheidung ist in diesem Fall nicht möglich. Werden zusätzliche Informationen wie z. B. relative Häufigkeit berücksichtigt, kann das unter Umständen zu einer

falschen Entscheidung führen. Dies wird auch durch die Evaluierung der Ergebnisse der wortbasierten Methode bei monolingualen Texten bestätigt (vgl. Kapitel 13). Werden gleich lautende Wörter aus allen Sprachmodellen entfernt, steht für die Spracherkennung noch weniger Informationen zur Verfügung. Außerdem erschwert es die Pflege der Sprachmodelle erheblich, da jedes Mal, wenn ein Sprachmodell hinzugefügt oder gelöscht wird, die Wortlisten angepasst werden müssen.

2) Die Methode eignet sich gut für die meisten westeuropäischen Sprachen. Für stark flektierende Sprachen erweist sie sich als ungeeignet. Die relativen Häufigkeiten der einzelnen Wörter in den Sprachmodellen der beiden von LangIdent unterstützten kyrillischen Sprachen Russisch und Ukrainisch sind im Vergleich zu den anderen Sprachen eher niedrig. Weder im Russischen noch im Ukrainischen gibt es die grammatische Klasse Artikel. Die Informationen über das Geschlecht, die Zahl und die Fallzugehörigkeit werden durch die Flexion ausgedrückt. Entsprechend niedrig ist die Wahrscheinlichkeit, dass in einem kurzen Text die in der Wortliste enthaltenen Wörter gefunden werden.

3) Der oben beschriebene Algorithmus ermöglicht es, längere Passagen in einer Sprache, eingebettet in eine andere Sprache, zu erkennen. Voraussetzung dafür ist, dass es hier um einen standardisierten Text handelt. Standardisiert bedeutet in diesem Fall, dass der Text vollständige Sätze enthält. Ein Beispiel dafür stellen Zeitungsartikel oder Belletristik dar. Ein Buchtitel oder ein kurzes Zitat, eine Auflistung der Eigennamen oder der Name einer Organisation sind Gegenbeispiele. Sie enthalten nicht unbedingt hoch frequente Wörter. Manchmal ist es jedoch wünschenswert, die Sprache solcher nicht-standardisierten Texte zu identifizieren. In solchen Fällen scheitert der wortbasierte Ansatz.

Da bei der Evaluierung der Performanz von LangIdent bei der monolingualen Analyse Bayes' sche Entscheidungsregel und Markov-Ketten die besten Ergebnisse erzielt haben, wurde dieses Klassifikationsmodell für die multilinguale Analyse ausgewählt.

Die Berechnung der Wahrscheinlichkeiten für alle Wörter aus dem Testdokument nimmt wesentlich mehr Zeit in Anspruch, als die Ermittlung der Anzahl der im Sprachmodell gefundenen Wörter des Testdokuments. Daher

musste der Algorithmus so angepasst werden, dass auch längere Dokumente in einer angemessenen Zeit analysiert werden können.
Nachfolgend wird der Algorithmus ausführlich beschrieben:
Das Testdokument wird anhand der Leerzeichen und Zeilenumbrüche in einzelne Wörter zerlegt. Jedes Wort wird normalisiert, d. h. es werden alle Zeichen entfernt, die keine Buchstaben oder Apostrophe darstellen, Großbuchstaben werden in Kleinbuchstaben umgewandelt, eventuell vorhandene Leerzeichen am Wortanfang und –ende werden ebenfalls eliminiert. Die normalisierte Wortform wird in einem Vector gespeichert. Gleichzeitig wird in einem anderen Vector die genaue Position des Wortes im Text abgelegt.
Es wird eine Hashtabelle erstellt, die alle Wörter aus dem Dokument in ihrer normalisierten Form mit Logarithmen ihrer Auftretenswahrscheinlichkeiten in jeder Sprache enthält. Der Logarithmus der Auftretenswahrscheinlichkeit eines Wortes in der gegebenen Sprache setzt sich aus den Logarithmen der Übergangswahrscheinlichkeiten der in dem Wort enthaltenen Tri-Gramme zusammen, die summiert werden. Die Berechnung der Übergangswahrscheinlichkeiten der Tri-Gramme wurde bereits in 3.3.2 und 9.2.2 ausführlich beschrieben und wird hier nicht näher erläutert. In der Hashtabelle wird im Gegensatz zum oben erwähnten Vector jedes Wort nur einmal gespeichert.
Anschließend werden für die ersten N Einträge (Elemente 0 bis N -1) im Vector die Auftretenswahrscheinlichkeiten in der Hashtabelle nachgeschlagen und summiert. Die Sprache mit der höchsten Wahrscheinlichkeit ist die aktuelle Sprache des Dokuments. Im nächsten Schritt werden die Elemente 1 bis N analysiert. Dieser Vorgang wird so lange wiederholt, bis das Dokumentende erreicht ist. Bekommt eine andere Sprache in einem Textabschnitt bzw. einem Intervall eine höhere Wahrscheinlichkeit als die aktuelle Sprache, wird angenommen, dass hier ein Sprachwechsel stattfindet. Die Position des ersten Wortes des betreffenden Abschnitts im Vector wird zusammen mit der identifizierten Sprache in einer weiteren Hashtabelle gespeichert.
Falls von dem Benutzer eine anschließende syntaktische Analyse gewünscht ist, wird in dem betreffenden Textabschnitt nach Auftreten bestimmter Indizien für

ein Sprachwechsel gesucht, um die Position des Sprachwechsels genau zu bestimmen.

Um die geeigneten Indizien zu ermitteln, wurden die authentischen multilingualen Dokumente aus dem Internet analysiert. Es wurde beobachtet, dass eine neue Sprache im Dokument am häufigsten mit folgenden syntaktischen Mitteln und Formatierungen eingeführt wird: Zeilenumbruch, doppelte und einfache Einführungsstriche, runde Klammern, Wechsel der Schriftart und/oder –farbe, Formatierungswechsel (fett, kursiv, unterstrichen).

Da von LangIdent in der aktuellen Version nur .txt-Format unterstützt wird, gehen bei der Konvertierung die meisten Formatierungsinformationen verloren. Aus diesem Grund werden im Moment bei der syntaktischen Analyse nur Zeilenumbrüche, runde Klammern und doppelte Einführungsstriche berücksichtigt. Die einfachen Einführungsstriche wurden ausgenommen, da sie in der Praxis sehr leicht mit Apostrophen verwechselt werden können.

Werden in dem Textabschnitt die oben erwähnten Merkmale gefunden, wird die Position des Sprachwechsels entsprechend korrigiert und in einer Baumstruktur gespeichert, die die Ausgabe ihrer Elemente in einer sortierten Reihenfolge erlaubt. Ansonsten wird die Position des ersten Wortes des Intervalls in dem Vector nachgeschlagen und in der Baumstruktur gespeichert.

Bei der Identifizierung der Sprachen in einem multilingualen Text ohne syntaktische Analyse konnte festgestellt werden, dass die Position des Sprachwechsels in der Regel um wenige Wörter nach links verschoben wurde. So ist bei einem Intervall von acht Wörtern, in dem die ersten zwei Wörter in einer Sprache sind, und die restlichen in einer anderen, die Wahrscheinlichkeit für die zweite Sprache höher. Als Position des Sprachwechsels wird jedoch die Position des ersten Wortes des Intervalls angenommen. Deswegen wurde der Algorithmus so abgeändert, dass die Position des Sprachwechsels um ein Viertel der Intervalllänge nach rechts verschoben wird.

Anschließend wird von dem System die Länge des Texts in jeder der identifizierten Sprachen zusammen gerechnet, der Anteil jeder Sprache an dem Gesamttext ermittelt und zusammen mit den genauen Positionen jeder Sprache im Text ausgegeben.

10 Beschreibung der Implementation

Im folgenden Kapitel wird gezeigt, wie die im letzten Kapitel vorgestellten Kernkomponenten von LangIdent in Java programmiert werden. Zu diesem Zweck wird das UML-Klassendiagramm erstellt und die wichtigsten Klassen beschrieben.

10.1 Programmiersprache und die Entwicklungsumgebung

Die Frage nach der Programmiersprache, in der das System implementiert werden sollte, war einfach zu beantworten. Java war die einzige Programmiersprache, die von den beiden Autoren der vorliegenden Arbeit in einem gewissen Maß beherrscht wird. Java eignet sich gleichermaßen gut für die Textverarbeitung und für die Programmierung anspruchsvoller grafischen Oberflächen. Der objektorientierte Charakter der Sprache erleichtert die Entwicklung und die Wartung der Programme. Das Entwicklungspaket für Java – JDK (Java Development Kit) – wird immer weiter entwickelt und enthält eine große Anzahl an fertigen Klassen, auf die der Programmierer zurückgreifen kann. Für die Entwicklung des Sprachidentifizierungssystems LangIdent wurde die aktuellste JDK-Version 1.5 (Tiger - Release) verwendet. Die wichtigste Erweiterung dieser Version sind die generischen Datentypen (Generics), die mehr Sicherheit bei der Arbeit mit den Datenstrukturen wie Vector, ArrayList u. ä. bieten.

Als Java Entwicklungsumgebung wurde Open-Source-Software *Eclipse* ausgewählt, da es den Entwicklern zahlreiche vorteilhafte Funktionalitäten wie Java-Editor, automatische Fehlererkennung, inkrementelles Kompilieren, Debuggen usw. anbietet. Dank diesen Funktionen ist es eine sehr komfortable und umfangreiche Umgebung zur Entwicklung von Java Anwendungen, die sich aber auch für andere Programmiersprachen erweitern lässt.

10.2 Klassenbeschreibung

Im Folgenden wird ein Überblick über die statische Struktur des Systems mittels eines UML-Klassendiagramms gegeben. In dem Diagramm werden die wichtigsten Klassen des Systems sowie deren Beziehungen untereinander abgebildet.

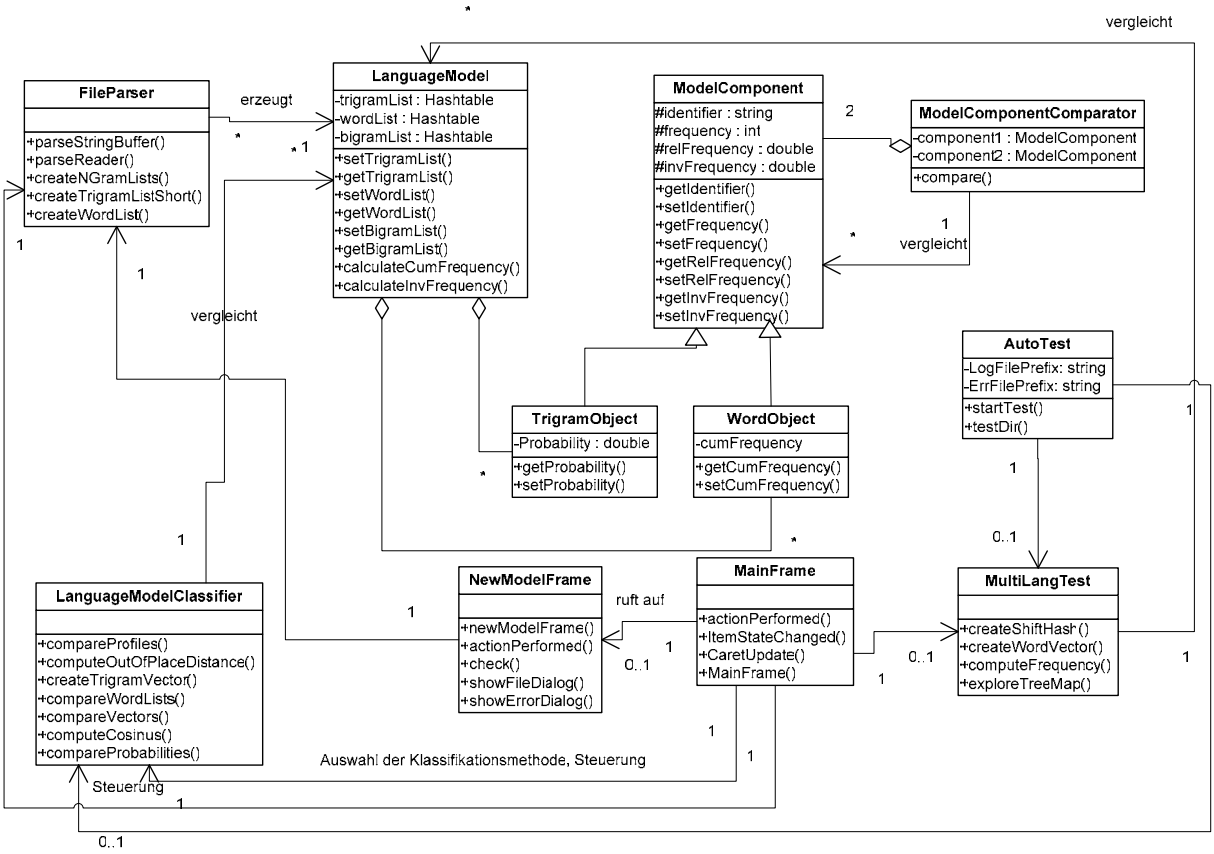


Abbildung 16: UML-Klassendiagramm

Die abstrakte Klasse *ModelComponent*, sowie die beiden Klassen *TrigramObject* und *WordObject* definieren elementare Bestandteile eines Sprachmodells. Diese Klassen werden im Punkt 10.2.1 detailliert beschrieben.

Die Klasse *LanguageModell* realisiert ein Sprachmodell, das dazu dient, Informationen über die Verteilung von Bi-, Tri-Grammen und Wörtern zu speichern. Sie wird im Punkt 10.2.2 näher betrachtet.

Die Klasse *FileParser* erzeugt Sprachmodelle anhand von Textdokumenten. Die Realisierung der zur Klasse gehörenden Methoden wird im Punkt 10.2.3 erläutert.

Die Klasse *LanguageModelClassifier* ordnet monolinguale Testdokumente der richtigen Sprache unter Anwendung des angegebenen Klassifikationsalgorithmus zu (s. Punkt 10.2.4).

Für die Bestimmung mehrerer Sprachen in einem Dokument wird die Klasse *MultiTest* benutzt. Eine ausführliche Beschreibung der Klasse ist im Punkt 10.2.5 gegeben.

Die Klassen *AutoTest* und *SplitText* realisieren eine Testumgebung für das Sprachidentifizierungssystem LangIdent. *SplitText* teilt ein Dokument in Abschnitte vorgegebener Größe. Die Klasse *AutoTest* ermöglicht die Erkennung der Sprache aller in einem Verzeichnis enthaltenen Dateien in Batch-Betrieb. Im Punkt 10.2.6 werden die beiden Klassen näher beschrieben.

Die Klassen *MainFrame*, *NewModelFrame, AutoTestFrame* und *SplitTextFrame* realisieren das Benutzer-Interface und starten den Modellbildungs- und den Sprachidentifikationsprozess (s. Punkt 10.2.7).

10.2.1 ModelComponent, TrigramObject, WordObject

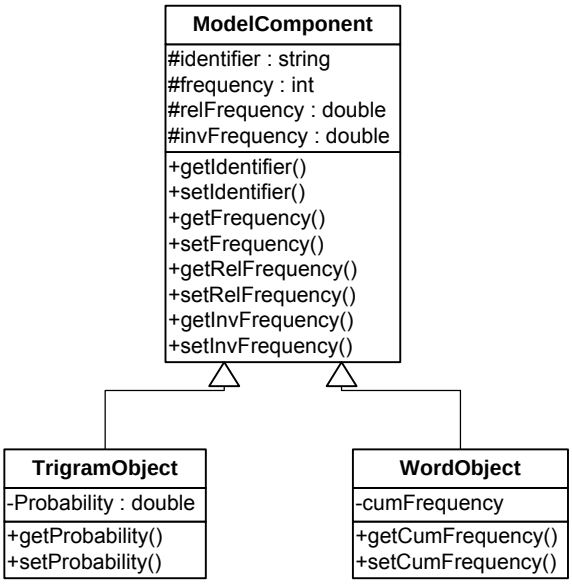


Abbildung 17: Klassen ModelComponent, TrigramObject, WordObject

Die Klasse *ModelComponent* ist eine abstrakte Oberklasse für Klassen, die elementare Spracheinheiten, wie Wörter oder N-Gramme definieren. Die Klasse beschreibt die gemeinsamen Eigenschaften aller Spracheinheiten, sowie die dazugehörigen Zugriffsmethoden.

Klasse: ModelComponent	
Attribute	**Beschreibung**
identifier : String	hier wird die Spracheinheit (z.B. ein Wort oder ein Tri-Gramm) gespeichert
frequency : int	absolute Häufigkeit der Einheit in einem Dokument
relFrequency : double	relative Häufigkeit der Einheit in einem Dokument
invFrequency : double	inverse Häufigkeit der Einheit in einem Dokument (vgl. J. M. Prager)
Methoden	**Beschreibung**
getProbability():double *setProbability(prob*:double*)*:void *setFrequency(freq*:int*)*:void …	Zugriffsmethoden, erlauben lesenden und schreibenden Zugriff auf Attribute

Die Klasse *TrigramObject* ist eine Spezialisierung der Klasse *ModelComponent*. Sie erweitert ihre Oberklasse um ein weiteres Attribut – die Übergangswahrscheinlichkeit für das Tri-Gramm in der von dem Sprachmodell repräsentierten Sprache (vgl. T.Dunning).

Klasse: TrigramObject	
Attribute	**Beschreibung**
probability : double	Übergangswahrscheinlichkeit
Methoden	**Beschreibung**
getProbability(): double *setProbability(prob*: double*)*:void	Zugriffsmethoden auf Attribut *probability*

Die im Sprachmodell enthaltenen Bi-Gramme stellen ebenfalls Instanzen der Klasse *TrigramObject* dar, da sie keine neuen Attribute besitzen, und es demnach keinen Grund gab, eine eigene Klasse für sie zu definieren.
Die Klasse *WordObject* ist eine weitere Spezialisierung der Klasse *ModelComponent*. Sie erweitert ihre Oberklasse um eine weitere statistische Größe – die kumulative Häufigkeit des Wortes in der Wortliste.

Klasse: WordObject	
Attribute	**Beschreibung**
cumFrequency : double	Kumulative Häufigkeit
Methoden	**Beschreibung**
getCumFrequency(): double *setCumFrequency(cF*:double*)*:void	Zugriffsmethoden auf Attribut *cumFrequency*

Falls das System um weitere Klassifikationsmethoden erweitert werden soll, die andere statistische Größen verwenden, können sowohl die Oberklasse als auch die von ihr abgeleiteten Unterklassen mit minimalem Aufwand entsprechend angepasst werden.

10.2.2 LanguageModel

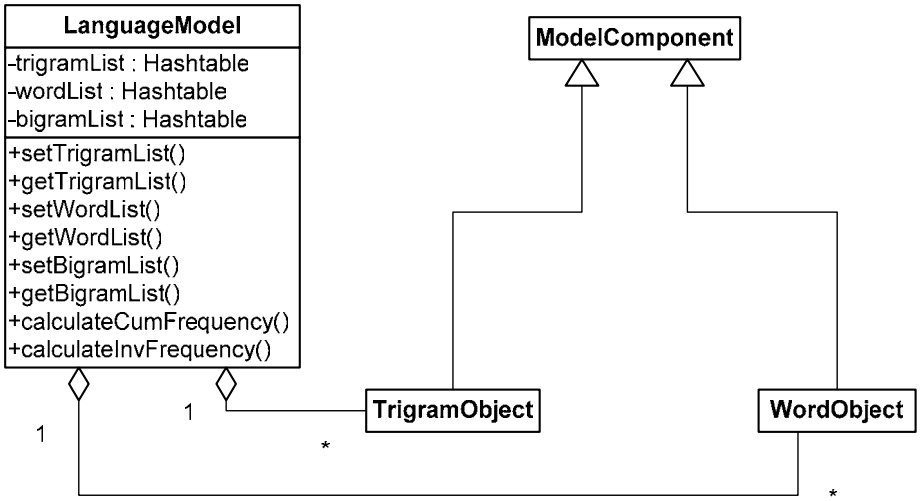


Abbildung 18: Klasse LanguageModel

Eine Instanz der Klasse *LanguageModel* beschreibt ein konkretes Sprachmodell bzw. Dokumentmodell. Sie enthält eine Liste von Tri-Grammen, eine Liste von Bi-Grammen, sowie eine Liste von Wörtern. Die Klasse *LanguageModel* bietet außerdem die Zugriffsmethoden für ihre Attribute sowie zwei Funktionen für die Berechnung von statistischen Größen – der kumulativen Häufigkeit und der inversen Dokumenthäufigkeit.

Klasse: LanguageModel	
Attribute	**Beschreibung**
name: String	Name des Sprachmodells (z. B. Deutsch)
alphabetSize: int	Größe des Alphabets
trigramList : Hashtable<String, TrigramObject>	Liste von Objekten der Klasse *TrigramObject*
bigramList : Hashtable<String, TrigramObject>	Liste von Objekten der Klasse *TrigramObject*
wordList : Hashtable<String, WordObject>	Liste von Objekten der Klasse *WordObject*
Methoden	**Beschreibung**
calculateCumFrequency(hash: Hashtable<String, WortObject>*)*: void	Berechnet die kumulative Häufigkeit für die Wortliste eines Sprachmodells
calculateInvFrequency(languageModels: Vector<LanguageModel>, *mode*: int*)*: void	Berechnet die inverse Dokumenthäufigkeit der Tri-Gramme (*mode* gleich eins) bzw. Wörter (*mode* gleich zwei) für alle im Vector enthaltenen Sprachmodelle
getTrigramList(): Hashtable<String, TrigramObject> *setTrigramList(tList*: Hashtable<String, TrigramObject>*)*: void …	Zugriffsmethoden auf die Klassenattribute

10.2.3 FileParser

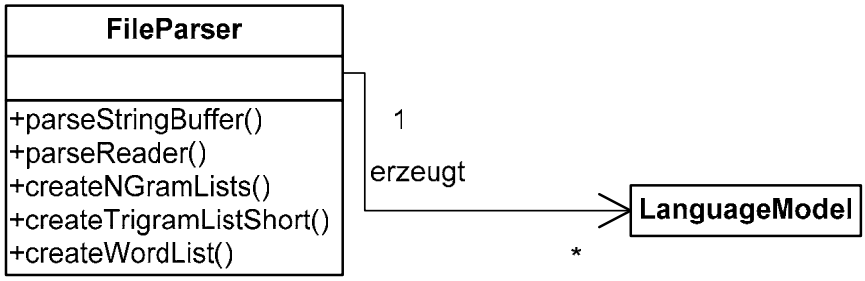


Abbildung 19: Klasse FileParser

Die Klasse *FileParser* wird für die Erstellung von Sprach- bzw. Dokumentmodellen verwendet. Ihre Instanz erzeugt für jedes Trainings- bzw. Testdokument eine Instanz der Klasse *LanguageModel.*

Klasse: FileParser	
Methoden	**Beschreibung**
parseStringBuffer(buf: StringBuffer*)*: void	Führt Normalisierung eines Textdokumentes durch - Großbuchstaben werden in Kleinbuchstaben umgewandelt - Zeichen, die keine Buchstaben und keine Apostrophe sind, werden durch Leerzeichen ersetzt - die Abstände zwischen zwei benachbarten Wörtern werden auf zwei Leerzeichen aufgefüllt
parseReader(br: BufferedReader): StringBuffer	Wandelt einen BufferedReader in einen StringBuffer um und ruft anschließend die Methode *parseStringBuffer()* auf, um den Text zu normalisieren.
createNGramLists(lm: LanguageModel, *buf*: StringBuffer, *choice*: int, *number*: int): void	Erzeugt die Bi- und Tri-Gramm-Listen für das übergebene Sprachmodell und berechnet die Alphabetgröße ausgehend vom normalisierten Text. Für die Bi-Gramme werden die absoluten Häufigkeiten ermittelt, für Tri-Gramme zusätzlich die relativen Häufigkeiten und die Übergangswahrscheinlichkeiten. Die Bi-Gramm Liste enthält alle Bi-Gramme aus dem Text. Der Parameter *choice* gibt an, welche Tri-Gramme in die Tri-Gramm Liste aufgenommen werden sollen: 1 - alle Tri-Gramme 2 - die n häufigsten Tri-Gramme 3 - Tri-Gramme, die häufiger als n Mal vorkommen n wird dabei durch den Parameter *number* bestimmt.
createTrigramListShort(buf: StringBuffer): Hashtable <String, TrigramObject>	Erstellt die Tri-Gramm-Liste für ein Dokumentmodell. Die Liste enthält alle im Dokument gefundenen Tri-Gramme, sowie deren absoluten und relativen Häufigkeiten.
createWordList(buf: StringBuffer, *choice*: int, *number*: int, *percent*: double): Hashtable<String, WordObject>	Erstellt die Wortliste für ein Sprach- oder ein Dokumentmodell, berechnet die absoluten und relativen Häufigkeiten der in der Liste enthaltenen Wörter. Der Parameter *choice* gibt die Option für die Wortlisteerstellung an: 1 – alle Wörter 2 – die n häufigsten Wörter 3 – die häufisten Wörter, deren kumulative Häufigkeit den durch den Parameter *percent* angegebenen Wert erreicht

	4 – alle Wörter, die nicht länger als n Zeichen sind. Der Parameter *number* kann je nach gewählter Option entweder die Anzahl der Wörter in der Wortliste oder die maximale Wortlänge bestimmen.

10.2.4 LanguageModelClassifier

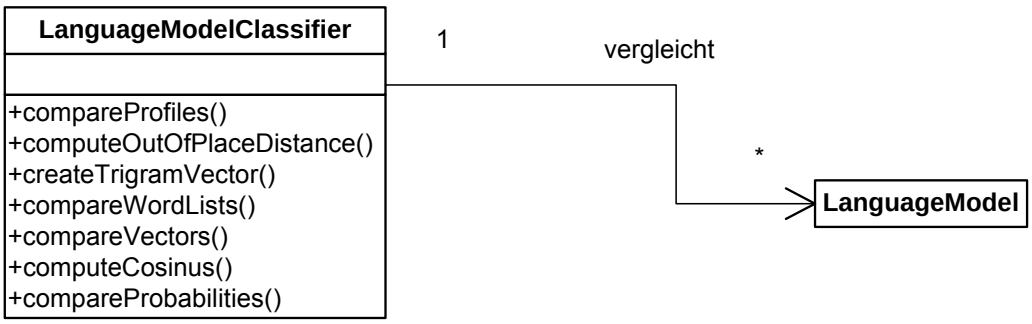


Abbildung 20: Klasse LanguageModelClassifier

Die Klasse *LanguageModelClassifier* realisiert mehrere Klassifikationsmethoden, die bei der Identifizierung der primären Sprache eines Testdokuments eingesetzt werden:

1. Das Ad Hoc Ranking Verfahren (out of place Methode)
2. Das Vektorraum Modell
3. Die Bayes'sche Entscheidungsregel
4. Die wortbasierte Methode.

Klasse: LanguageModelClassifier	
Methoden	**Beschreibung**
compareProfiles(lm: LanguageModel, *languageModels*: Vector<LanguageModel>): int	Ermittelt den Index des Sprachmodells, das die kleinste Out-of-place Distanz zum Testdokumentmodell besitzt
computeOutOfPlaceDistance(a: Vector<String>, *b*: Vector<String>*)*: int	Berechnet die Out-of-place Distanz zwischen einem Sprachmodell und einem Dokumentmodell. Die beiden Modelle werden in diesem Fall auf die sortierten Listen von

	Tri-Grammen reduziert.
createTrigramVector(a:ArrayList <TrigramObject>*)*: Vector<String>	Überführt einen ArrayList, der aus einer Hashtabelle erstellt wurde, in einen Vector. Da die Tri-Gramme sowohl in den Sprach- als auch in den Dokumentmodellen unsortiert vorliegen, ist dieser Schritt notwendig, um sie nach ihrer absoluten Häufigkeit zu ordnen. In dem Vector werden nur die Tri-Gramme selbst gespeichert.
compareWordLists(buf: StringBuffer, *languageModels*: Vector <LanguageModel>*)*: int	Liefert die Position des Sprachmodells, in dessen Wortliste die meisten Wörter aus dem Testdokument gefunden wurden. Weisen mehrere Sprachmodelle die gleiche Trefferanzahl auf, wird das Sprachmodell mit der höchsten Summe der relativen Häufigkeiten der gefundenen Wörter ausgewählt.
compareVectors(lm: LanguageModel, *languageModels*: Vector<LanguageModel>*)*: int	Ermittelt den Index des Sprachmodells, dessen Vektor die kleinste Kosinusdistanz zum Vektor des Testdokuments besitzt.
computeCosinus(lm1: LanguageModel, *lm2*: LanguageModel*)*: double	Berechnet die Kosinusdistanz zwischen einem Sprachmodellvektor und einem Dokumentmodellvektor.
compareProbabilities(buf: StringBuffer, *languageModels*: Vector<LanguageModel>*)*: int	Liefert die Positions des Sprachmodells mit der höchsten Wahrscheinlichkeit, dass der übergebene Text in der vom Sprachmodell repräsentierten Sprache verfasst ist, zurück (Bayes'sche Entscheidungsregel)

Außerdem enthält die Klasse einige Hilfsmethoden zur Ermittlung der Position des größten oder des kleinsten Elements in einem Array mit Ganz- oder Dezimalzahlen. Sie werden hier nicht näher erläutert.

10.2.5 MultiLangTest

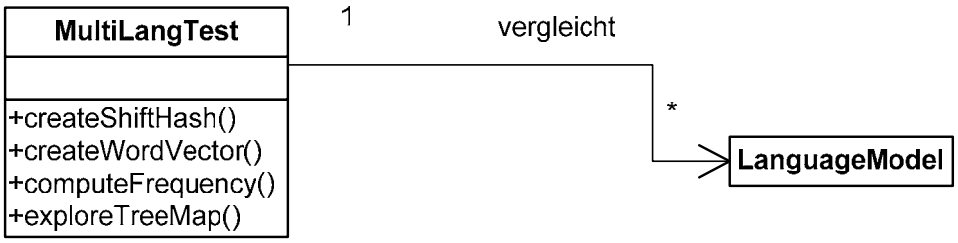


Abbildung 21: Klasse MultiLangTest

Die Klasse MultiLangTest enthält Methoden, die bei der Identifizierung von mehreren Sprachen in einem Dokument verwendet werden.

Klasse: MultiLangTest	
Methoden	**Beschreibung**
createShiftHash(languageModels: Vector<LanguageModel>, *buf*: StringBuffer, *interval*: int, *synt*: boolean): TreeMap<Integer, Integer>	Erstellt eine Hashtabelle, in der die Positionen des Sprachwechsels zusammen mit der dazugehörigen Sprache gespeichert werden. Der Parameter *interval* bestimmt, wie viele Wörter auf einmal ausgelesen werden (vgl. Punkt 9.3 S. 77), der boolische Wert *synt* gibt an, ob anschließend eine syntaktische Analyse durchgeführt werden soll.
createWordVector(words: Vector<String>, *languageModels*: Vector<LanguageModel>*)*: Vector<Hashtable>	Erstellt einen Vector, der für jede Sprache aus dem Vector *languageModels* eine Hashtabelle enthält. In den einzelnen Hashtabellen werden die Wahrscheinlichkeiten für jedes Wort aus dem Vector *words* gespeichert.
computeFrequency(word: String, *lm*: LanguageModel*)*: double	Berechnet die Wahrscheinlichkeit für das Wort *word*, dass es in der durch das Sprachmodell *lm* repräsentierten Sprache verfasst ist.
exploreTreeMap(tree: TreeMap<Integer, Integer>*)*: TreeMap<Integer, Integer>	Die Baumstruktur, die die Positionen des Sprachwechsels und die jeweiligen Sprachen enthält, wird analysiert. Nach der durchgeführten syntaktischen Analyse der Abschnitte, wo ein Sprachwechsel angenommen wurde,

	kann es vorkommen, dass zwei nacheinander folgende Elemente der Baumstruktur die gleiche Sprache enthalten. In so einem Fall wird das zweite Element gelöscht. Außerdem werden die Elemente eliminiert, bei denen die Differenz zwischen der eigenen Position und der Position des nächsten Elements weniger als 12 beträgt.

10.2.6 SplitText, AutoTest

Mit den Klassen *SplitText* und *AutoTest* wird eine Testumgebung für das System LangIdent realisiert. Die genaue Beschreibung der Testreihe zusammen mit den erzielten Ergebnissen ist im Kapitel 13 zu finden.

Eine Instanz der Klasse *SplitText* teilt eine Datei oder alle in einem Verzeichnis enthaltenen Dateien in kleinere Dateien vorgegebener Größe. So wurden beispielsweise für den Testkorpus Dateien von 50, 100, 250, 500 und 1000 Byte erstellt. Damit die automatisch erstellten Texte im nächsten Schritt auch automatisch getestet und ausgewertet werden können, sollten die Dateien nach einem bestimmten Muster benannt werden. Der Dateiname soll einen Unterstrich enthalten, gefolgt von dem Namen des entsprechenden Sprachmodells oder dessen Abkürzung, z.B. spiegel_de.txt, lemonde_franz.txt u. s. w.

Klasse: SplitText	
Methoden	**Beschreibung**
testDir(file: File, *targetDir*: File, *sizeArray*: int[]*)*: void	Eine Instanz der Klasse *File* kann eine Referenz auf eine Datei oder ein Verzeichnis sein. Falls *file* eine Datei referenziert, wird die Methode *parseFile()* aufgerufen, ansonsten wird es rekursiv über alle Dokumente im Verzeichnis *file* sowie in allen seinen Unterverzeichnissen iteriert und dabei die Methode *parseFile()* für jedes Dokument aufgerufen. Der Parameter *targetDir* gibt ein Verzeichnis an, wo die erstellten Dateien abgelegt werden sollen, *sizeArray* enthält die Größen der zu erstellenden Dateien in Byte.
parseFile(file: File, *targetDir*:	Zuerst wird ein Ordner in dem Zielverzeichnis

File, *sizeArray*: int[]): void	angelegt. Der Name des Ordners stimmt mit dem Teil des Dateinamens nach dem Unterstrich überein. Für jede Größe aus dem Array *sizeArray* wird ein eigenes Verzeichnis erstellt, dessen Name sich aus dem Namen des Oberverzeichnisses und der Größe zusammensetzt (z. B. de50). Die Datei wird in Abschnitte der jeweiligen Größe geteilt, die in dem entsprechenden Ordner gespeichert werden. Wurden die Ausgangsdateien nach der oben erwähnten Konvention genannt, kann ausgehend von Ordnernamen auf die Sprache der erstellten Dateien geschlossen werden

Die Klasse *AutoTest* realisiert die eigentliche Testprozedur. Sie führt Sprachidentifikation aller im ausgewählten Verzeichnis enthaltenen Dokumente.

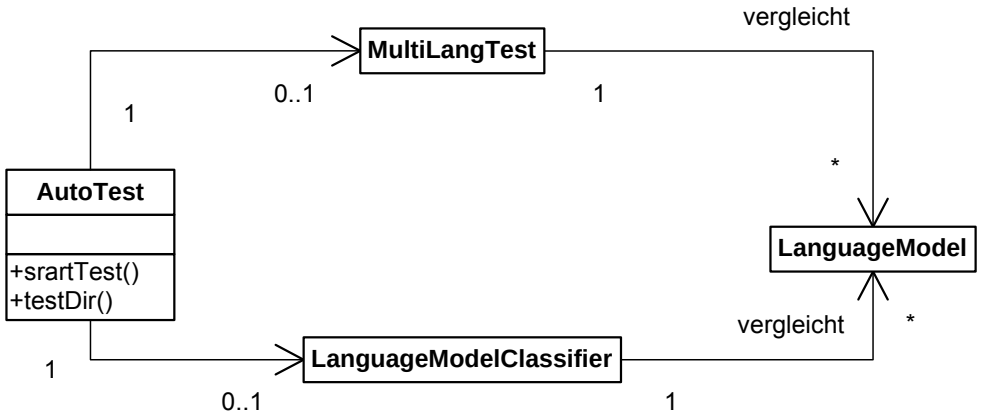


Abbildung 22: Klasse AutoTest

Der Benutzer kann festlegen, ob die primäre Sprache oder alle im Dokument enthaltenen Sprachen identifiziert werden sollen. Des Weiteren kann bei der monolingualen Analyse das Klassifikationsverfahren ausgewählt werden. Bei der Erkennung aller Sprachen des Dokuments ist die Angabe des Intervalls notwendig. Die Testergebnisse werden in den so genannten Log-Dateien festgehalten, für jedes Verzeichnis wird eine eigene Log-Datei erstellt. Alle Klassifikationsfehler bei der Erkennung der primären Sprache des Dokuments

werden in den Error-Dateien zusammengefasst. Auch hier wird für jedes Verzeichnis eine eigene Error-Datei angelegt, um die Fehleranalyse zu erleichtern. Anschließend werden alle Fehler in einer weiteren Datei erfasst, differenziert nach der Sprache, der Dokumentlänge und der verwendeten Klassifikationsmethode.

Klasse: AutoTest	
Attribute	**Beschreibung**
LogFilePrefix: String	Präfix der zu erstellenden Log-Dateien
ErrFilePrefix: String	Präfix der zu erstellenden Error-Dateien
Methoden	**Beschreibung**
startTest(startDir: File, *models*: File, *choice*: int, *interval*: int*)*: void	Initialisiert je eine Instanz der Klasse *PrintStream* für die Log-Datei und Error-Datei sowie eine Instanz der Klasse *PrintWriter* für die GesamtError-Datei, lädt die Sprachmodelle aus der durch den Parameter *models* referenzierten Datei in einen Vector, initialisiert einen leeren Array für die Testergebnisse und ruft anschließend die Methode *testDir()*, um die eigentliche Testprozedur zu starten. Der Parameter *choice* gibt an, welche Methode für den Identifikationsprozess eingesetzt werden soll: 1 - das Vektorraum Modell 2 - das Out-of-place Verfahren 3 - die Bayes'sche Entscheidungsregel 4 - der wortbasierte Vergleich 5 - alle Klassifikationsmethoden 6 - multilinguale Analyse. Im Fall der multilingualen Analyse bestimmt der Parameter *interval* die Intervalllänge bzw. die Fenstergröße.
testDir(file: File, *lm*: Vector<LanguageModel>, *choice*: int, *interval*: int, *ps*: PrintStream, *psErr*: PrintStream, *falseRes*: int[], *os*: PrintWriter): void	Testet rekursiv alle in dem durch den Parameter *file* referenzierten Verzeichnis enthaltenen Dateien mit der durch *choice* angegebenen Methode. Die Testergebnisse werden in die Log-Dateien geschrieben, die falsch klassifizierten Dateien werden außerdem in den Err-Dateien registriert.

Dass die beiden Klassen AutoTest und SplitText nicht die gewünschte Flexibilität bei der Benennung der zu teilenden Dateien, bei der Auswahl des Zielverzeichnisses für die Testergebnisse u. s. w. aufweisen, hängt damit

zusammen, dass sie in erster Linie für den eigenen Bedarf entwickelt wurden. Später wurden sie in das Hauptprogramm integriert, um den Benutzer bei der Erstellung von Testkorpora und der Evaluierung der Testergebnisse zu unterstützen.

10.2.7 MainFrame, NewModelFrame, SplitTextFrame, AutoTestFrame

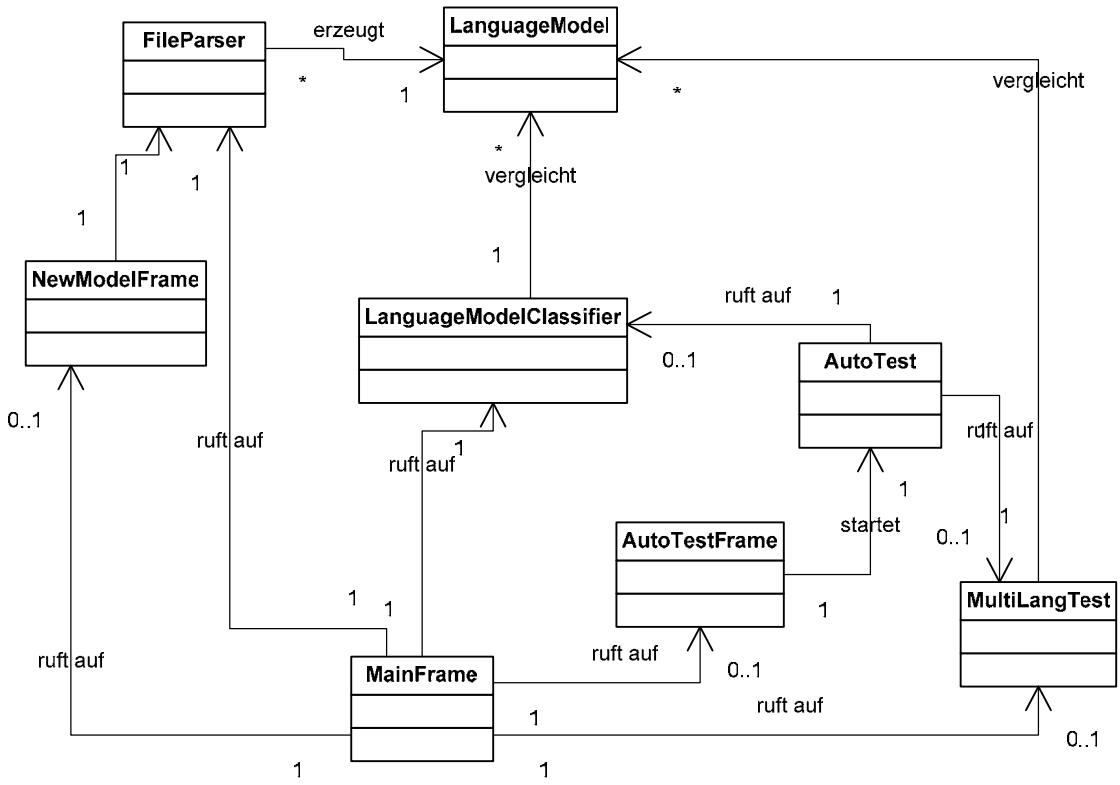


Abbildung 23: Klassen MainFrame, NewModelFrame

Die Klassen *MainFrame, NewModelFrame, SplitTextFrame und AutoTestFrame* realisieren die graphische Benutzerschnittstelle.

Die Klasse *MainFrame* realisiert das Hauptfenster des Programms, seine beiden Ansichten: die Textansicht und die Sprachmodellansicht. In der Textansicht hat der Nutzer die Möglichkeit zur Eingabe des zu identifizierenden Texts und zur Auswahl der Klassifikationsmethoden. In der Sprachmodellansicht können die einzelnen Sprachmodelle angesehen, gelöscht und bearbeitet werden,

außerdem können die Sprachmodelllisten geladen und gespeichert werden. Alle anderen visuellen Klassen werden ebenfalls von der Klasse MainFrame aufgerufen.

Wegen der großen Anzahl von Attributen und Methoden werden sie hier nicht einzeln beschrieben. Die meisten Attribute stellen grafische Komponenten wie Schaltflächen, Radiobuttons, Auswahllisten oder Menüeinträge dar. Die anderen definieren verschiedene Stile für die Darstellung des Textes oder bestimmen das Layout des Hauptfensters.

Die Klasse *MainFrame* implementiert die Interfaces *ActionListener, ItemListener* und *CaretListener,* deswegen muss die Klasse die Methoden enthalten, die beschreiben, wie das Programm auf die durch die grafischen Komponenten ausgelösten Ereignisse reagieren soll. Die Methode *actionPerformed()* wird aufgerufen, falls ein *ActionEvent* von einer Schaltfläche oder einem Radiobutton ausgelöst wird. Wird in der Auswahlliste ein neuer Eintrag ausgewählt, liegt ein *ItemEvent* vor, es wird demnach die Methode *ItemStateChanged()* aufgerufen. Beim Bewegen des Kursors tritt ein *CaretEvent* auf, das von der Methode *CaretUpdate()* behandelt wird. Die anderen Methoden dienen zum Aufbau oder zur Aktualisierung der grafischen Komponenten.

Die Klasse *NewModelFrame* ermöglicht es dem Benutzer, die Parameter für die Sprachmodellerstellung einzugeben, und startet anschließend den Sprachmodellbildungsprozess.

Klasse: NewModelFrame	
Attribute	**Beschreibung**
tFile, tName, tTrigramNumber, tTimesNumber, tWordNumber, tCutOff, tWordLength: JTextField	Textfelder für die Eingabe der einzelnen Parameter für die Sprachmodellerstellung
names: Vector<String>	Vector mit den Namen der bereits in der Liste enthaltenen Sprachmodelle
rAllTrigrams, rNTrigrams, rNTimesTrigrams, rAllWords, rNWords, rCutOff, rShortWords: JRadioButton	RadioButtons für die Auswahl der für die Sprachidentifikation einzusetzenden Methode.
Methoden	**Beschreibung**
newModelFrame(modelNames: Vector<String>, *mf*: MainFrame)	Konstruktor, initialisiert das Fenster. Erhält als Parameter einen Vector mit den Namen der Sprachmodelle, sowie die Instanz der

	Klasse MainFrame, die den Konstruktor aufgerufen hat.
actionPerformed(e: ActionEvent*)*: void	Beschreibt, wie die einzelnen von den Schaltflächen und Radiobuttons ausgelösten Ereignisse behandelt werden sollen.
check(): boolean	Überprüft die Benutzerangaben auf ihre Vollständigkeit und Korrektheit: - der Name des zu erstellenden Sprachmodells muss angegeben werden und eindeutig sein - die Datei mit den Trainingsdaten muss ausgewählt sein - die Anzahl der Tri-Gramme oder der Wörter, sowie die maximale Wortlänge und die minimale absolute Häufigkeit der Tri-Gramme müssen positive Ganzzahlen sein - die kumulative Häufigkeit muss eine positive Dezimalzahl sein
showFileDialog(mode: String*)*: File	Öffnet einen Dialog zur Dateiauswahl. Der Parameter *mode* gibt an, ob die Datei zum Lesen (open) oder zum Speichern (save) geöffnet wird.
showErrorDialog(error: String*)*: void	Gibt die durch den Parameter *error* definierte Fehlermeldung aus.

Die Klassen *SplitTextFrame* und *AutoTestFrame* ermöglichen es dem Benutzer, die Parameter für die Erstellung der Dateien bzw. für die Durchführung der Tests einzugeben. Da der Aufbau der beiden Klassen dem der oben beschriebenen Klasse *NewModelFrame* sehr ähnlich ist, werden die einzelnen Attribute und Methoden nicht näher erläutert.

11 Probleme

Um einen Eindruck zu vermitteln, mit welchen Problemen die Autoren im Laufe der Implementierungsphase konfrontiert wurden, werden einige von ihnen im folgenden Abschnitt skizziert.

Da das Sprachidentifizierungssystem LangIdent Sprachen mit unterschiedlichen Zeichensätzen unterstützen sollte, musste sichergestellt werden, dass auch die Zeichen, die nicht im lateinischen/englischen Alphabet enthalten sind, korrekt ausgelesen, dargestellt und gespeichert werden können. In einem mehrsprachigen Kontext bietet es sich an, als Kodierung Unicode zu benutzen, da sich in Unicode auch landesspezifische Zeichen darstellen lassen.

Java benutzt auf Quelltextebene zwar den 16-Bit-Unicode-Zeichensatz, beim Zugriff auf das Dateisystem werden jedoch die lokalen Einstellungen des Betriebssystems übernommen. Java bietet die Möglichkeit, einen String in eine andere Kodierung zu überführen. Die Ausgangs- und die Zielkodierung müssen angegeben werden. In den meisten Fällen ist die Ausgangskodierung aber nicht bekannt.

Die Aufgabe der Kodierungserkennung ist keinesfalls trivial. Um die Kodierung eines Textes zu bestimmen, muss die Verteilung von verschiedenen Zeichen und Steuer-Sequenzen untersucht werden. Manchmal werden dazu auch Listen der hoch frequenten Wörter einer Sprache in verschiedenen Kodierungen herangezogen. Die Bewältigung dieser Aufgabe erforderte aber mehr Zeit, als den Autoren zur Verfügung stand. Deswegen wurde das Problem umgegangen, indem vorausgesetzt wurde, dass sowohl die Trainingsdokumente als auch die zu identifizierenden Texte in der Kodierung Unicode Big Endian vorliegen. In diesem Fall kann der Konvertierungsschritt entfallen, es muss lediglich bei der Initialisierung einer Instanz der Klasse *InputStreamReader* (beim Lesen aus einer Datei) oder *OutputStreamReader* (beim Schreiben in eine Datei) die verwendete Kodierung angegeben werden, z.B.:

```
BufferedReader is = new BufferedReader(new InputStreamReader(
        new FileInputStream(fileName), "UnicodeBig"));
```

Sprachmodellenanzeige

Am Anfang der Implementierungsphase, als noch keinen Trainingskorpus existierte, wurden für die Erstellung von Sprachmodellen relativ kurze Texte verwendet. Die erstellten Sprachmodelle wurden nicht zur Sprachidentifizierung eingesetzt, sondern dienten lediglich zum Testen der Modellerstellungsroutine und der Methoden zum Anzeigen der Sprachmodellkomponenten – der Tri-Gramm-Liste und der Wortliste. Um die Darstellung einigermaßen lesbar zu machen, wurde auf den Gebrauch von der Klasse *JTextArea* verzichtet, die eine Komponente zur Darstellung eines formatierten mehrzeiligen Textes bereitstellt. Stattdessen wurde die Klasse *JEditorPane* benutzt, die auch HTML und RTF darstellen kann. Mit Hilfe von HTML konnten die Tabellen übersichtlich gestaltet werden.

Das Problem trat auf, als die Sprachmodelle anhand der Texte aus dem Trainingskorpus erzeugt wurden. Da die Trainingstexte etwa 200 KByte lang waren, enthielten die Tri-Gramm- und Wortlisten bis zu 5000 Elemente, was zu sehr langen Ladezeiten führte. Eine mögliche Lösung bestand darin, die beiden Listen parallel zu laden. Java unterstützt das Multithreadingkonzept, es wäre also problemlos realisierbar und würde die Ladezeiten um den Faktor 2 verkürzen. Das wäre aber immer noch nicht genug, um von einem verbesserten Bedienkomfort sprechen zu können. Eine weitere Möglichkeit wäre, dem Benutzer nur die ersten 200 Elemente anzuzeigen. Manchmal ist es aber durchaus interessant zu erfahren, welche Tri-Gramme und Wörter eher selten in einer Sprache vorkommen. Das kann insbesondere bei der Fehleranalyse behilflich sein, da z.B. bei der Berechnung der inversen Dokumenthäufigkeit allein das Vorhandensein des Elements in der Liste, nicht aber seine Häufigkeit entscheidend ist.

Die Autoren haben sich für die folgende Lösung entschieden: es werden nur die ersten 100 Elemente angezeigt, die beiden Anzeigebereiche werden jedoch mit Navigationspfeilen versehen, mit deren Hilfe der Benutzer in den Listen vor und zurück blättern kann.

Ein weiteres Problem bereitete die Bestimmung der Position der einzelnen Wörter im Text. Bei der Identifizierung aller Sprachen eines Dokuments sollte die Position der einzelnen Sprachen im Text möglichst genau bestimmt werden. Um

den im Kapitel 9.3 beschriebenen Algorithmus verwenden zu können, sollte der Text in einzelne Wörter geteilt werden.

Zum Zerlegen eines Strings in einzelne Tokens wird in Java normalerweise die Klasse *StringTokenizer* eingesetzt. Es ist möglich, zu bestimmen, anhand welcher Zeichen die Trennung erfolgen soll. Werden beim Aufruf des Konstruktors keine Trennzeichen (engl. delimiters) angegeben, wird der String anhand von Leerzeichen, Zeilenumbrüchen und Tabulatoren segmentiert. Die Klasse erlaubt es, über die Liste der Tokens zu iterieren, bietet jedoch keine Möglichkeit, die Position der einzelnen Tokens im Text zu bestimmen.

Eine etwas bessere Alternative bietet die Klasse *BreakIterator*, die es ermöglicht, logische Grenzen in einem Text zu erkennen, d.h. den Text in einzelne Zeichen, Wörter oder Sätze zu teilen. Auch die Position jedes Elements kann ermittelt werden.

Diese Klasse ist sehr nützlich bei der Textverarbeitung, da sie den Programmierer von der Notwendigkeit befreit, eigene Methoden zur Behandlung von Sonder- und Satzzeichen zu schreiben. Allerdings braucht sie Informationen über die Sprache, in der der Text geschrieben ist, ist bei einem Sprachidentifizierungssystem dementsprechend nutzlos.

Seit der Version 1.4 enthält JDK das Paket java.util.regex, das die Verwendung von regulären Ausdrücken in Java ermöglicht. Mit Hilfe von regulären Ausdrücken kann der Text nach bestimmten Mustern durchgesucht werden. Reguläre Ausdrücke stellen ein mächtiges Instrument zur Textverarbeitung dar, erfordern jedoch eine gewisse Einarbeitungszeit. Die *Pattern*-API enthält allerdings eine Anzahl von vordefinierten Character Klassen für häufig verwendete reguläre Ausdrücke. Für die Texttokenisierung wurde der Ausdruck „\s“ benutzt, der für die sogenannten Whitespaces (engl. whitespace characters) steht, wie z. B. Leerzeichen, Zeilenumbruch, Tabulator u. ä. Die Klasse *Matcher* stellt mehrere Methoden, mit denen die Treffer, die bei der Suche mit regulären Ausdrücken gefunden wurden, manipuliert werden können. Unter anderem ist auch die Bestimmung der Position der einzelnen Treffer in dem Text möglich.

12 Bedienung des Programms

Bevor das Programm zum ersten Mal gestartet wird, muss sichergestellt werden, dass auf dem System JDK 1.5 installiert ist.
Beim Starten des Programms werden Modelle von derzeit acht unterstützten Sprachen geladen, so dass die Identifizierung der Sprache sofort möglich ist.

Texteingabe

Der zu identifizierende Text kann in den Eingabebereich (links) eingetippt werden. Über das Menü DATEI/DATEI ÖFFNEN oder Klicken auf die Schaltfläche „Datei auswählen“ (alternativ mit der Abkürzungstaste <Strg>+<O>) wird das Dialogfenster zum Auswählen einer Datei aufgerufen. Nachdem die Auswahl durch einfaches Betätigen der Schaltfläche „Öffnen“ bestätigt wird, wird der Inhalt der Datei in den Eingabebereich geladen.

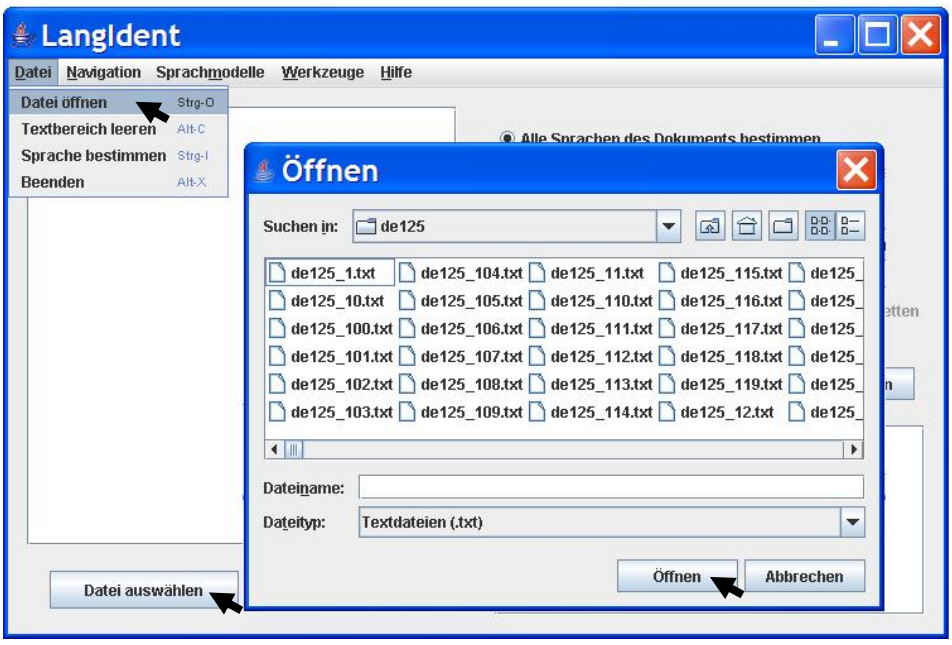


Abbildung 24: Benutzeroberfläche: Texteingabe

Es ist darauf zu achten, dass das Programm in der aktuellen Version nur Dateien im txt-Format unterstützt. Versucht der Benutzer eine Datei in einem anderen Format zu öffnen, bekommt er eine Fehlermeldung.

Abbildung 25: Benutzeroberfläche: Fehlermeldung bei der Auswahl einer Datei In einem nicht unterstützten Format

Durch Anklicken der Schaltfläche „Textbereich leeren“ oder durch Anwählen des Menüpunktes MENÜ/TEXTBEREICH LEEREN bzw. mit dem Tastenkürzel <Alt>+<C> kann der eingegebene Text gelöscht werden.

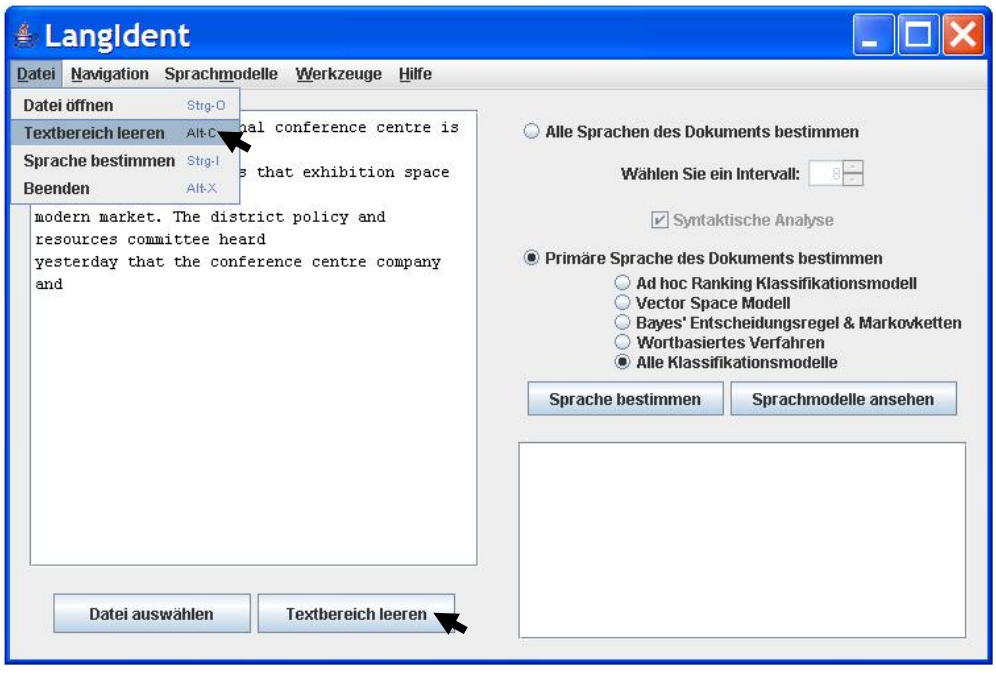


Abbildung 26: Benutzeroberfläche: Leeren des Textbereichs

Auswahl der Klassifikationsmethode und Starten des Sprachidentifikationsvorgangs

Bevor der Identifikationsprozess gestartet wird, kann der Benutzer auswählen, ob alle Sprachen des Dokuments oder nur die primäre Sprache bestimmt werden sollen.
Im Fall der multilingualen Analyse muss das Intervall bzw. die Fensterbreite angegeben werden. Es gibt an, wie viele Wörter in einem Schritt analysiert werden. Ist das Intervall zu klein, ist es möglich, dass das System nicht genug Informationen hat, um die Sprache des Textabschnitts korrekt zu identifizieren. Ist es dagegen zu groß, werden möglicherweise einige wenige Wörter in einer Sprache, die von einer anderen Sprache umgeben sind, unbemerkt bleiben. Sollen beispielsweise bereits drei nacheinander folgende englische Wörter in einem deutschen Text erkannt werden, sollte das Intervall auf fünf gesetzt werden: 5 = 2 * n -1, wobei *n* für die Anzahl der zu erkennenden Wörter steht.

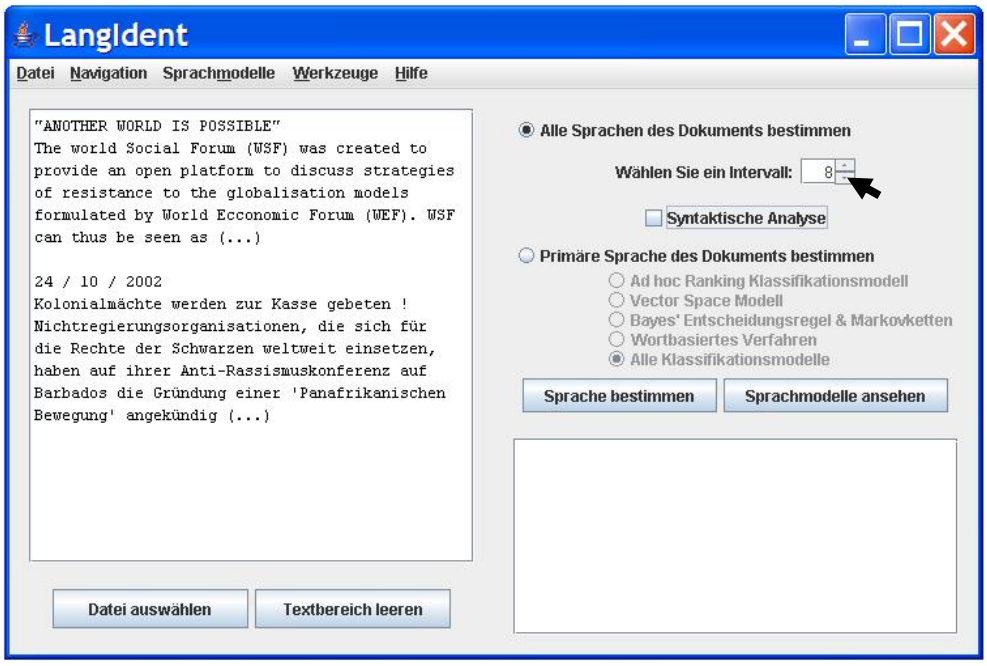


Abbildung 27: Benutzeroberfläche: Bestimmung des Intervalls

Zusätzlich kann festgelegt werden, ob anschließend eine syntaktische Analyse durchgeführt werden soll, die es unter Umständen erlaubt, anhand von

syntaktischen Merkmalen die Position des Sprachwechsels genau zu bestimmen.

Entscheidet sich der Benutzer für die Identifikation der primären Sprache des Dokuments, muss er festlegen, welche der vier Klassifikationsmethoden eingesetzt werden soll: Ad Hoc Ranking, Vector Space Modell, Bayes'sche Entscheidungsregel, wortbasiertes Verfahren oder alle vier Methoden zusammen.

Es gibt mehrere Möglichkeiten, den Identifikationsprozess zu starten: entweder durch Anwählen des Menüpunktes DATEI/SPRACHE BESTIMMEN oder durch Anklicken der Schaltfläche mit der gleich lautenden Beschriftung oder mit dem Tastenkürzel <Strg> + <I>.

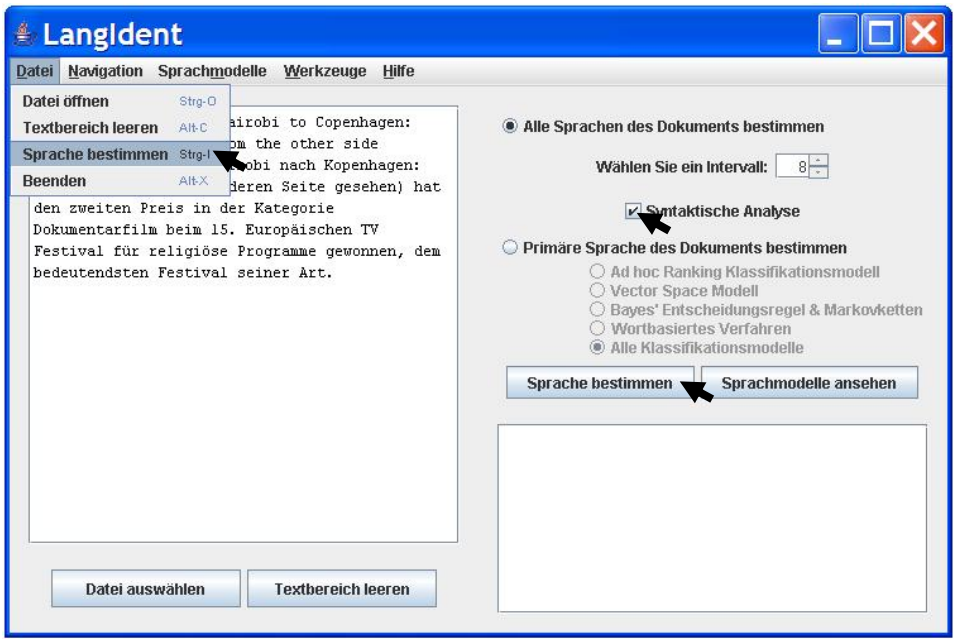


Abbildung 28: Benutzeroberfläche: Starten des Sprachidentifikationsprozesses

Das Ergebnis wird anschließend in dem Textbereich unten rechts angezeigt.

Im Fall der multilingualen Analyse wird angegeben, wie viel Prozent jede Sprache an dem Gesamttext ausmacht. Es folgt die Auflistung der Positionen, an denen ein Sprachwechsel stattgefunden hat, mit der Angabe der aktuellen Sprache. Die verschiedenen Sprachen werden in jeweils einer anderen Schriftart dargestellt.

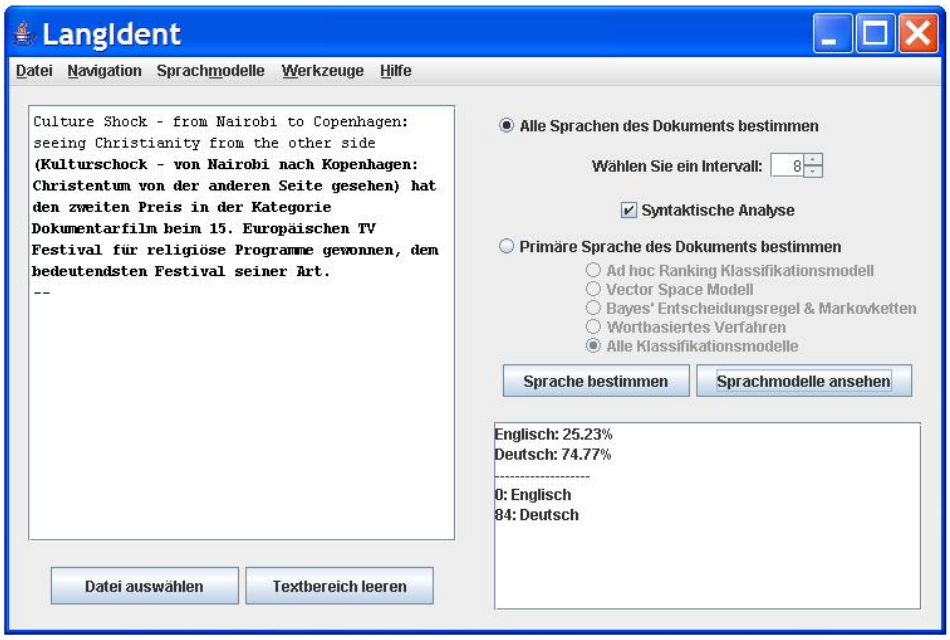


Abbildung 29: Benutzeroberfläche: Ausgabe der Ergebnisse bei der Sprachidentifikation multilingualer Texte

Wurde nur die primäre Sprache des Dokuments bestimmt, zeigt das Programm die identifizierte Sprache unter Angabe der angewandten Klassifikationsmethode. Wenn vom Benutzer die Option „alle Klassifikationsmodelle“ ausgewählt wurde, werden vier Ergebnisse differenziert nach der Klassifikationsmethode ausgegeben.

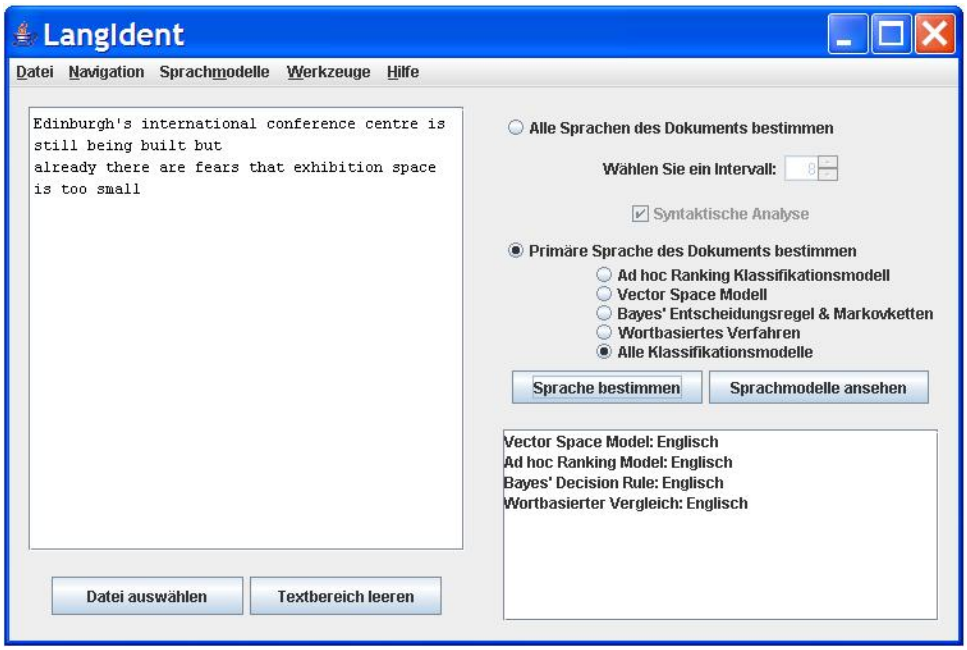


Abbildung 30: Benutzeroberfläche: Ausgabe der Ergebnisse bei der Identifikation der primären Sprache

Navigation durch die Sprachmodellliste, Bearbeitung von Sprachmodellen

Durch Anklicken der Schaltfläche „Sprachmodelle ansehen“ bzw. über den Menüeintrag NAVIGATION/SPRACHMODELLE ANZEIGEN (alternativ mit <Strg>+<M>) kann zu der nächsten Seite gewechselt werden. Hier werden die Sprachmodelle angezeigt.

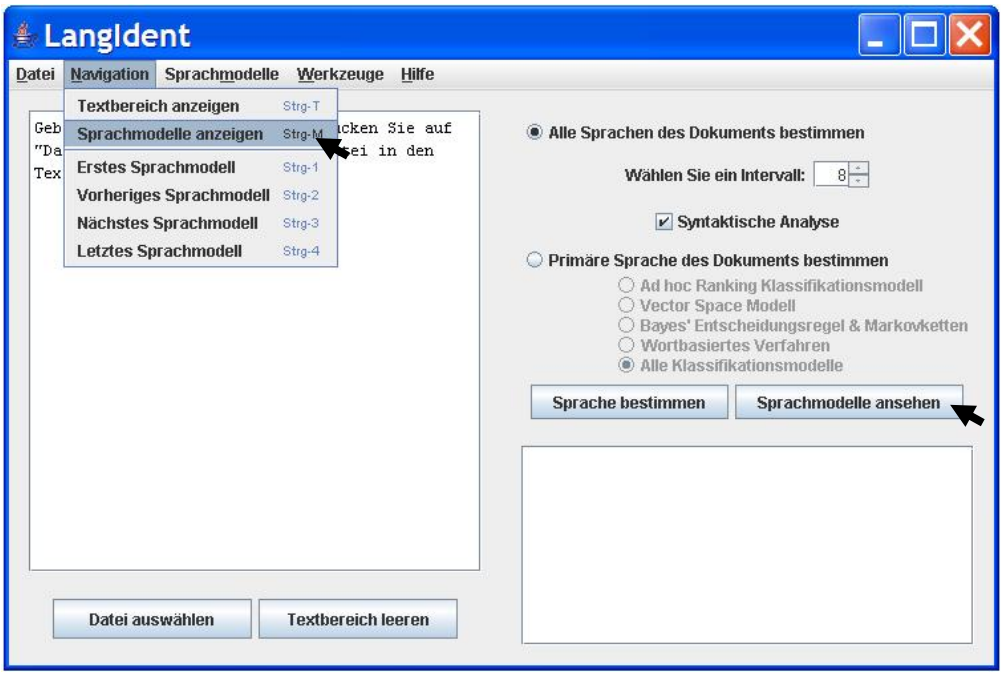


Abbildung 31: Benutzeroberfläche: Wechsel Sprachmodellenansicht

Beim Programmstart wird das erste Sprachmodell aus der alphabetisch sortierten Liste angezeigt. Die Navigation durch die Sprachmodellliste erfolgt entweder über die Pfeile rechts und links neben dem Auswahlfeld oder über die Menüpunkte NAVIGATION/ERSTES SPRACHMODELL, NAVIGATION/VORHERIGES SPRACHMODELL, NAVIGATION/NÄCHSTES SPRACHMODELL, NAVIGATION/LETZTES SPRACHMODELL. Die entsprechenden Abkürzungstasten sind respektive <Alt>+<1>, <Alt>+<2>, <Alt>+<3> und <Alt>+<4>.

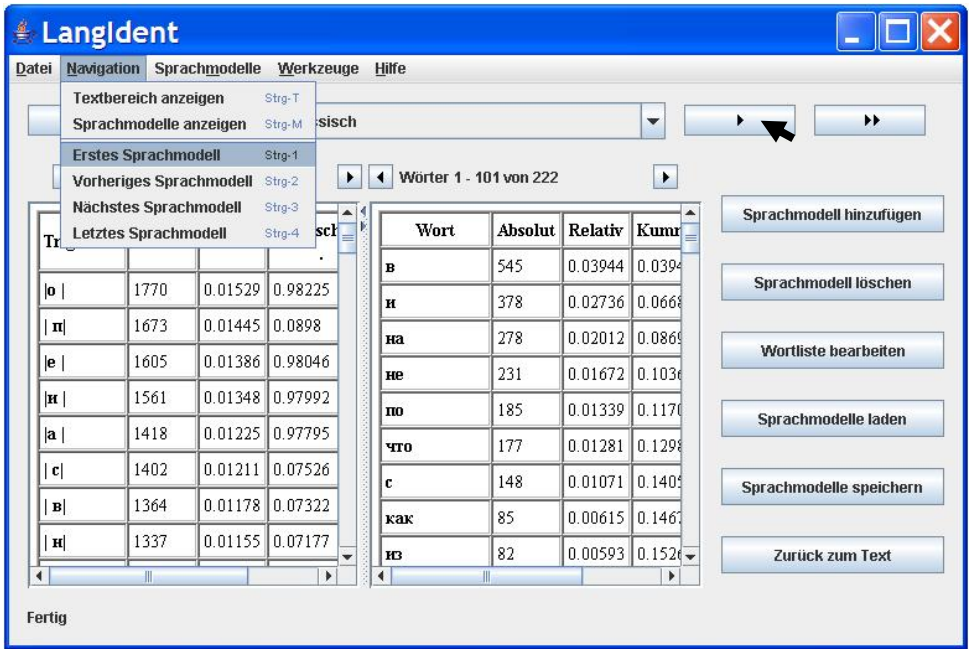


Abbildung 32: Benutzeroberfläche: Navigation durch die Sprachmodellliste

Die andere Möglichkeit, sich das gewünschte Sprachmodell anzeigen zu lassen, besteht darin, den Namen des Modells in der Auswahlliste anzuklicken.

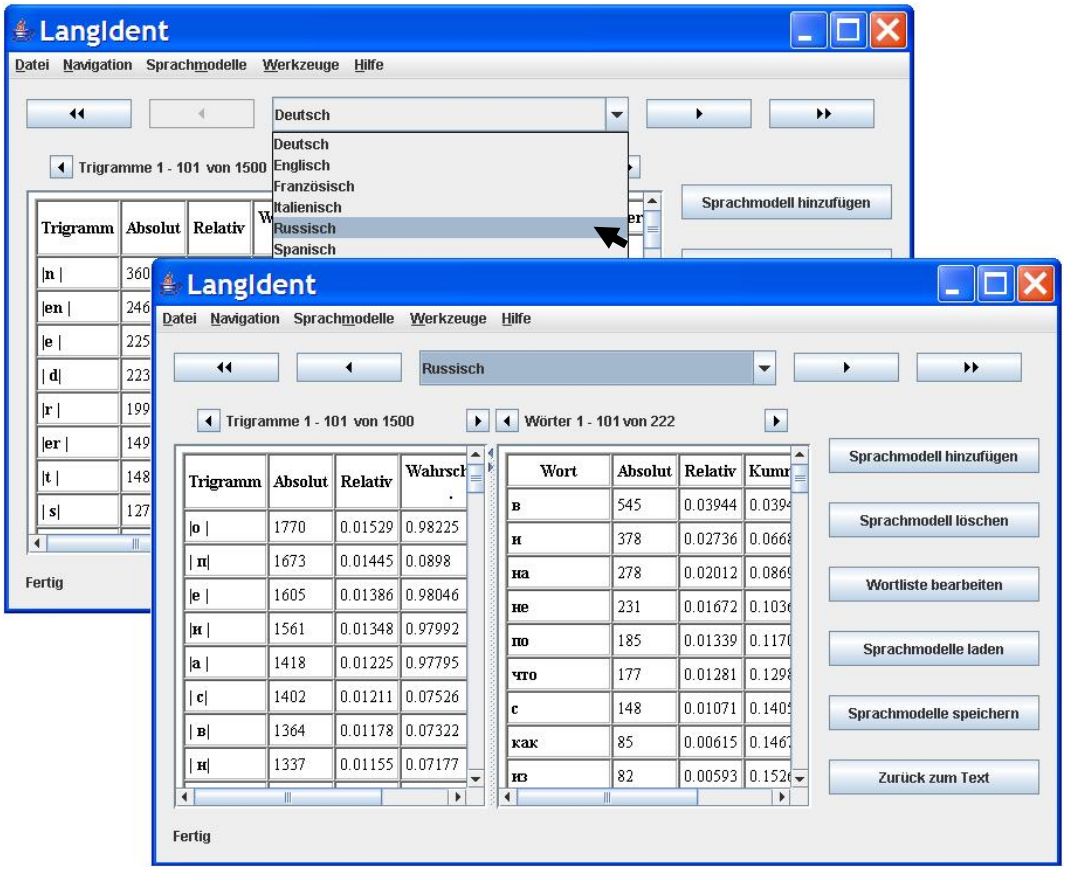


Abbildung 33: Benutzeroberfläche: Auswahl des Sprachmodells

In dem Textbereich links findet der Benutzer die Tri-Gramm-Liste des jeweiligen Sprachmodells. Je nach dem, welche Optionen bei der Modellerstellung gewählt wurden, kann sie entweder alle Tri-Gramme aus den Trainingsdokumenten in dieser Sprache, oder die N häufigsten Tri-Gramme enthalten, oder alle Tri-Gramme, die in den Trainingsdokumenten öfter als N Mal vorkommen, wobei N eine natürliche Zahl ist. Die Tri-Gramme sind absteigend nach ihrer absoluten Häufigkeit sortiert. Neben den absoluten Häufigkeiten werden auch ihre relativen Häufigkeiten, Übergangswahrscheinlichkeiten und inversen Dokument-häufigkeiten angezeigt.

Im rechten Textbereich wird die Wortliste des aktuellen Sprachmodells ausgegeben. Sie setzt sich zusammen entweder aus allen Wörtern aus den Trainingsdokumenten, oder aus den N häufigsten Wörtern, oder den häufigsten Wörtern, deren kumulative Häufigkeit eine angegebene Grenze erreicht, oder aus den Wörtern, die nicht länger als N Zeichen sind. Die Wortliste enthält Angaben über die absoluten, relativen und kumulativen Häufigkeiten der Wörter sowie ihre inversen Dokumenthäufigkeiten.

Da die Listen, insbesondere der Tri-Gramme, ziemlich groß sein können, abhängig von der Größe der Trainingsdokumente sowie der gewählten Einstellungen bei der Sprachmodellerstellung, kann es lange dauern, bis sie vollständig geladen sind. Um die Ladezeiten zu verkürzen, werden zuerst die ersten 100 Tri-Gramme bzw. Wörter ausgegeben. Mit Hilfe von oberhalb der Textbereiche platzierten Navigationselementen können die gesamten Listen angesehen werden.

Wegen der begrenzten Fenstergröße können die beiden Listen nicht in ihrer vollen Breite angezeigt werden. Durch Anklicken der kleinen Pfeile zwischen den beiden Textbereichen lässt sich ein Teilfenster über den ganzen Textbereich ausweiten, während das zweite verborgen bleibt. Die Liste kann jetzt vollständig angezeigt werden, das lästige horizontale Scrollen kann auf diese Weise vermieden werden. Das wiederholte Anklicken der Pfeile stellt den ursprünglichen Zustand wieder her.

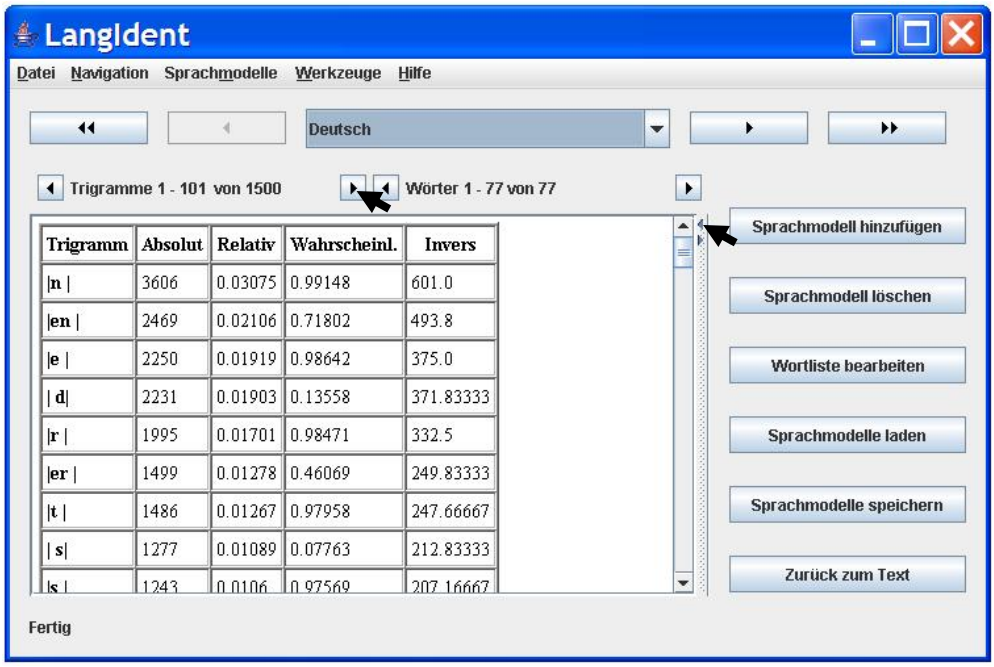


Abbildung 34: Benutzeroberfläche: Ansehen der Tri-Gramm-Liste

Die Wortliste eines Sprachmodells kann intellektuell nachbearbeitet werden. Durch Betätigen der Schaltfläche „Wortliste bearbeiten“ oder Anwählen des Menüs SPRACHMODELLE/WORTLISTE BEARBEITEN (<Strg>+<E>) öffnet sich ein Dialogfenster, wo alle Wörter der Liste angezeigt werden, diesmal alphabetisch geordnet, um das Auffinden bestimmter Wörter zu erleichtern.

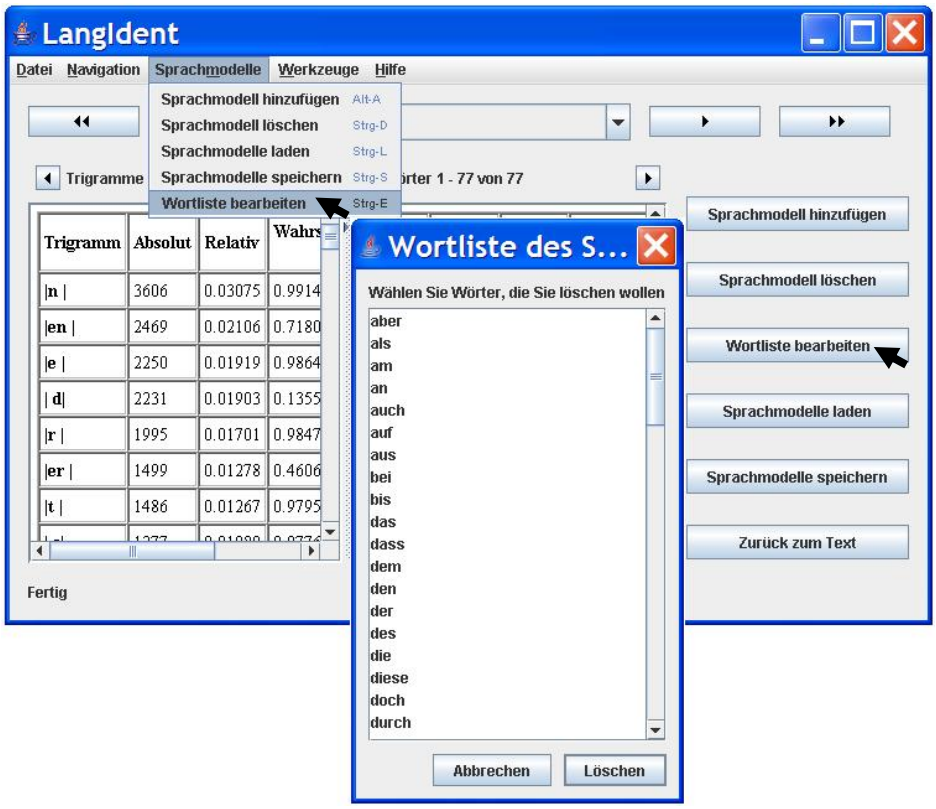


Abbildung 35: Benutzeroberfläche: Bearbeitung der Wortliste

Jetzt können die einzelnen Wörter gelöscht werden. Drücken und Halten der <Strg>-Taste ermöglicht das Auswählen mehrerer Einträge. Durch Anklicken der Schaltfläche „Löschen“ werden die ausgewählten Wörter aus der Wortliste entfernt, und anschließend die inversen Dokumenthäufigkeiten für die Wortlisten aller Modelle sowie die kumulative Häufigkeit für die im aktuellen Modell enthaltenen Wörter neu berechnet.

Hinzufügen neuer Sprachmodelle / Löschen von Sprachmodellen

Es können zu jedem Zeitpunkt neue Sprachmodelle hinzugefügt werden. Einfaches Betätigen der Schaltfläche „Sprachmodell hinzufügen“ (alternativ Anwählen des Menüs SPRACHMODELLE/SPRACHMODELL HINZUFÜGEN oder <Alt>+<A>) öffnet ein Dialogfenster, wo die Einstellungen für die Sprachmodellbildung vorgenommen werden können.

Der Benutzer soll einen Namen für das Sprachmodell eingeben, der eindeutig sein muss. Enthält die Sprachmodellliste bereits ein Sprachmodell mit dem gleichen Namen (Groß-/Kleinschreibung wird nicht berücksichtigt), wird der Benutzer aufgefordert, einen anderen Namen einzugeben.

Es wird festgelegt, welche Tri-Gramme in die Tri-Gramm-Liste aufgenommen werden sollen: ob *alle* in Trainingsdokumenten enthaltenen Tri-Gramme, oder nur die *N häufigsten* Tri-Gramme oder alle Tri-Gramme, die in den Trainingsdokumenten *öfter als n-mal vorkommen*.

Bei der Erstellung der Wortliste kann der Benutzer unter vier Möglichkeiten auswählen: *alle* aus den Trainingsdaten gewonnenen Wörter, die *N häufigsten* Wörter, die häufigsten Wörter, die *eine bestimmte kumulative Häufigkeit erreichen* oder die Wörter, die eine bestimmte *Wortlänge* nicht überschreiten.

Es wird nun die Datei ausgewählt, die das Trainingsmaterial enthält. Durch Anklicken der Schaltfläche „Modell erstellen“ werden die Eingaben bestätigt und der Prozess der Sprachmodellbildung gestartet. Das erstellte Sprachmodell wird zur Sprachmodellliste hinzugefügt. Die inversen Dokumenthäufigkeiten werden für alle Sprachmodelle neu berechnet.

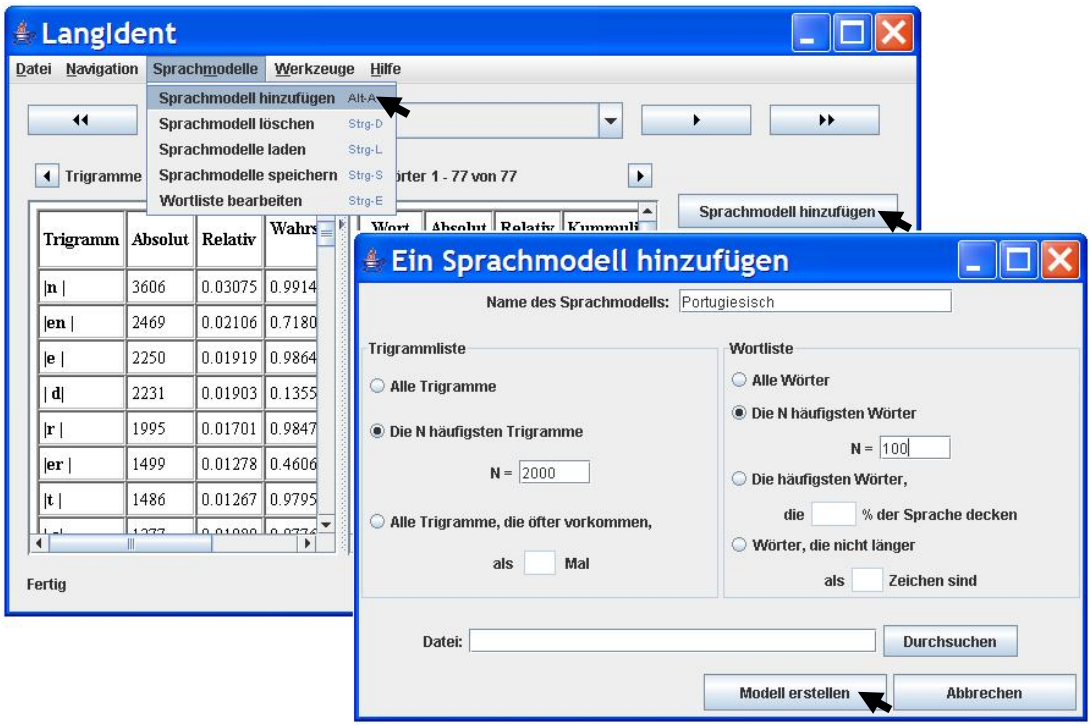


Abbildung 36: Benutzeroberfläche: Erstellen eines neuen Sprachmodells

Über die Schaltfläche „Sprachmodell löschen“ und über den Menüeintrag SPRACHMODELLE/SPRACHMODELL LÖSCHEN kann das aktuell angezeigte Sprachmodell gelöscht werden. Das Dialog zum Bestätigen des Löschvorgangs lässt sich auch mit der Abkürzungstaste <Strg>+<D> aufrufen.

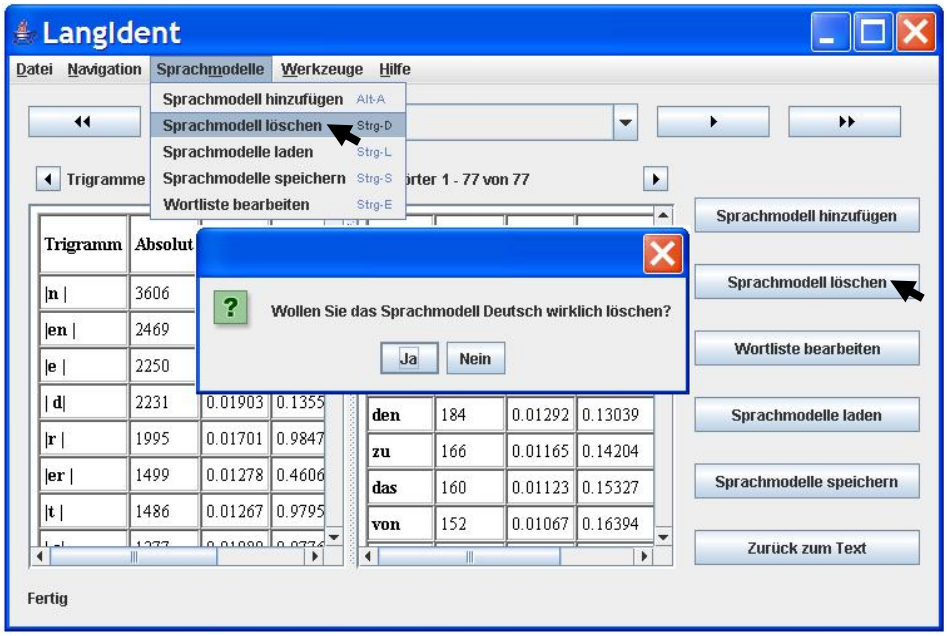


Abbildung 37: Benutzeroberfläche: Löschen des Sprachmodells

Nachdem das Sprachmodell aus der Liste entfernt wurde, werden für die Tri-Gramm- und Wortlisten aller Sprachmodelle die inversen Dokumenthäufigkeiten neu berechnet.

Laden und Speichern von Sprachmodellen

Über die Schaltfläche „Sprachmodelle laden“ bzw. über das Menü SPRACHMODELLE/SPRACHMODELLE LADEN kann jederzeit eine neue Sprachmodellliste in den Arbeitsbereich geladen werden.

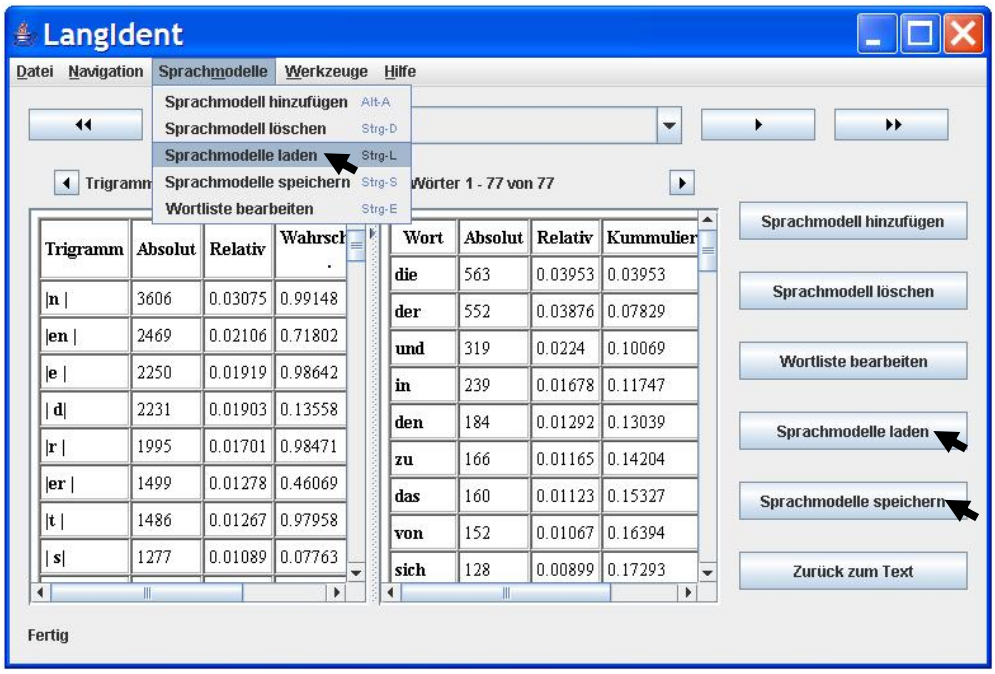


Abbildung 38: Benutzeroberfläche: Laden, Speichern von Sprachmodellen

Die aktuelle Sprachmodellliste wird nicht um neue Sprachmodelle erweitert, sie wird durch die neue Sprachmodellliste ersetzt.

Durch Betätigen der Schaltfläche „Sprachmodelle speichern“ ist es möglich, die Sprachmodelle in einer Datei zu sichern. Diese Funktion erlaubt es, mit verschiedenen Sprachmodellen zu arbeiten, ohne sie jedes Mal neu erstellen zu müssen.

Anwendung von SplitText

Über den Menüpunkt WERKZEUGE/SPLITTEXT (alternativ mit Abkürzungstaste <Alt>+<S>) wird ein Hilfsprogramm aufgerufen, dessen Aufgabe darin besteht, einen langen Text in Abschnitte vorgegebener Größe zu teilen.

Der Benutzer wählt die zu teilende Datei bzw. ein Verzeichnis, das eine oder mehrere Dateien in verschiedenen Sprachen enthält. Der Name der Datei sollte einen Unterstrich enthalten, gefolgt von dem Namen des Sprachmodells oder einer Abkürzung dessen, das mit der Sprache des Dokuments assoziiert wird (z.B. Spiegel_deutsch.txt oder Spiegel_de.txt).

Außerdem soll die Größe der zu erstellenden Dateien in Byte angegeben werden. Mehrere Angaben werden durch ein Semikolon getrennt.
Der Benutzer soll angeben, wo die erstellten Dateien gespeichert werden sollen.
Mit dem Betätigen der Schaltfläche „OK“ werden die Eingaben bestätigt.

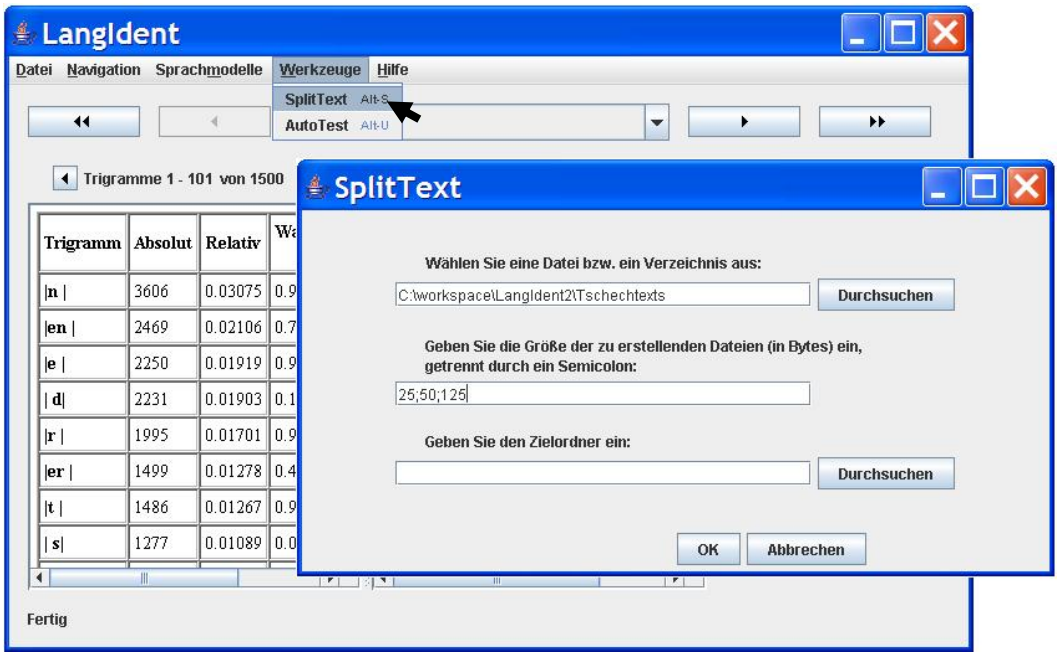


Abbildung 39: Benutzeroberfläche: Anwendung von SplitText

Es wird eine Verzeichnisstruktur angelegt, die mittels folgenden Schemas veranschaulicht werden kann:

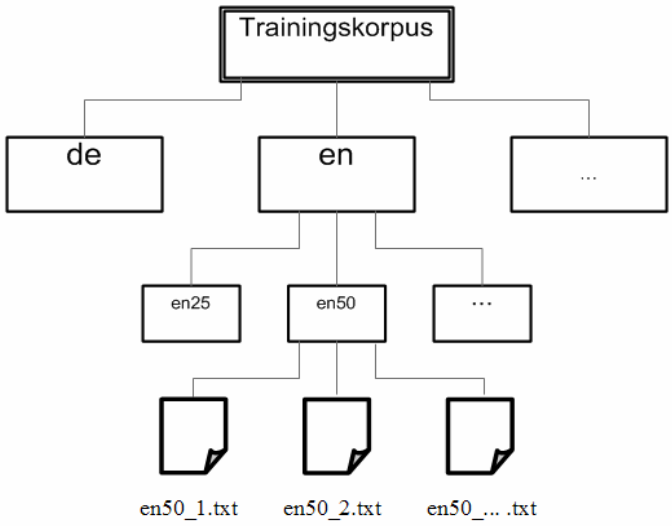


Abbildung 40: Schematische Darstellung der von SplitText angelegten Verzeichnisstruktur

Da die Teilung der Datei nur an den Wortgrenzen erfolgt, kann die Größe der erstellten Dateien um bis zu 50 Byte von der Benutzervorgabe abweichen.

Anwendung von AutoTest und Auswertung der Testergebnisse

Müssen große Datenmengen getestet werden, erweist sich das Werkzeug AutoTest als sehr nützlich. Es befindet sich unter dem Menüpunkt „Werkzeuge/AutoTest“, kann aber auch mit dem Tastenkürzel <Alt>+<U> gestartet werden.
Der Benutzer wählt das Verzeichnis aus, das die zu testenden Dateien enthält. Die Datei mit den Sprachmodellen muss ebenfalls angegeben werden.
Es wird festgelegt, ob nur die primäre Sprache des Dokuments oder alle Sprachen identifiziert werden sollen.
Im Falle der monolingualen Analyse kann der Benutzer bestimmen, welche Methode zu Sprachidentifikation eingesetzt werden soll: Ad Hoc Ranking, Vector Space Modell, Bayes’sche Entscheidungsregel, wortbasiertes Verfahren oder alle vier Methoden zusammen.
Entscheidet sich der Benutzer für die multilinguale Analyse, muss das Intervall vorgegeben werden.
Nachdem die Eingaben durch Betätigen der Schaltfläche „OK“ bestätigt wurden, bestimmt das Programm die Sprache aller im ausgewählten Verzeichnis oder seinen Unterverzeichnissen enthaltenen Dateien.

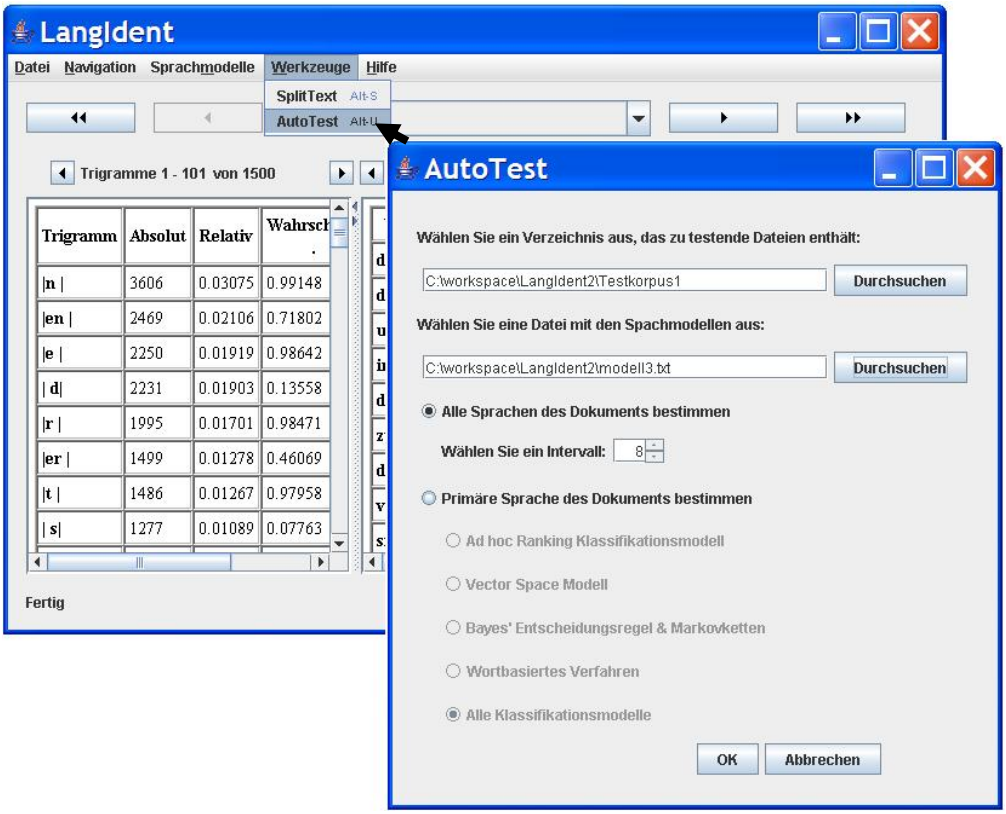


Abbildung 41: Benutzeroberfläche: Anwendung von AutoTest

Die Ergebnisse werden in so genannten Log-Dateien festgehalten, für jedes Verzeichnis wird eine eigene Log-Datei angelegt. Zu jeder Datei werden folgende Angaben gemacht: der absolute Pfad, die verwendete Methode und das Ergebnis.

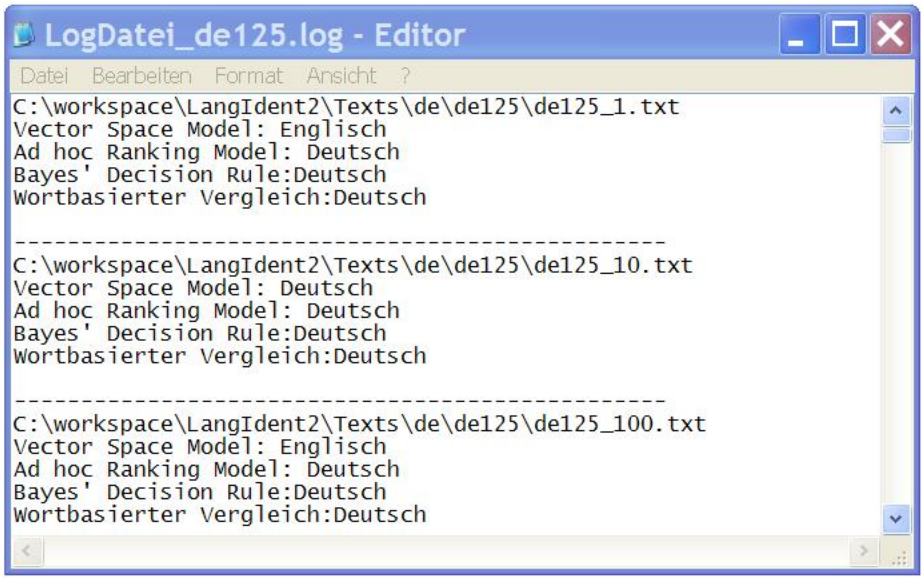


Abbildung 42: Auszug aus einer LogDatei

Wenn nur die primäre Sprache des Dokuments bestimmt werden soll, wird zusätzlich für jedes Verzeichnis eine Error-Datei angelegt. Sie enthält Angaben zu den Dateien, deren Sprache nicht korrekt identifiziert wurde.

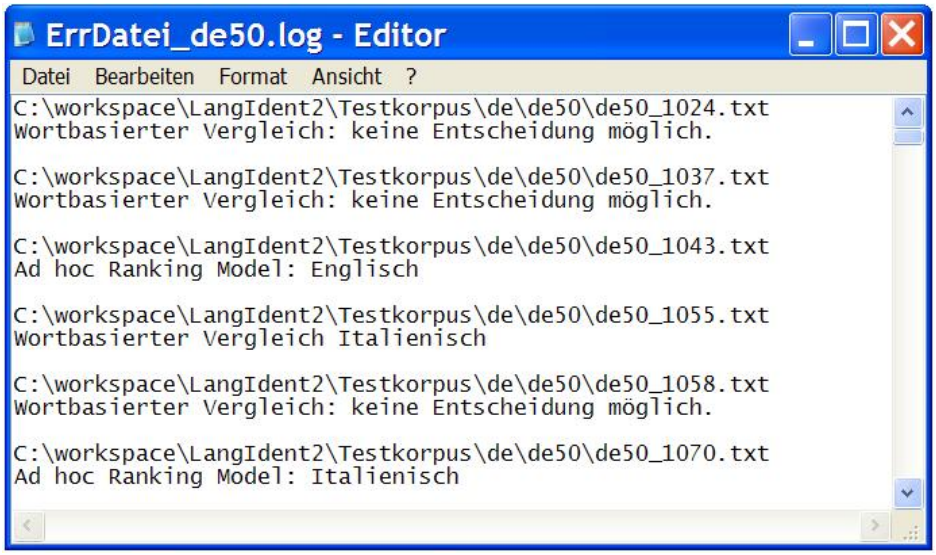


Abbildung 43: Auszug aus einer Error-Datei (1)

Außerdem wird in der Datei ErrGesamt.log eine Zusammenfassung der Testergebnisse angeführt.

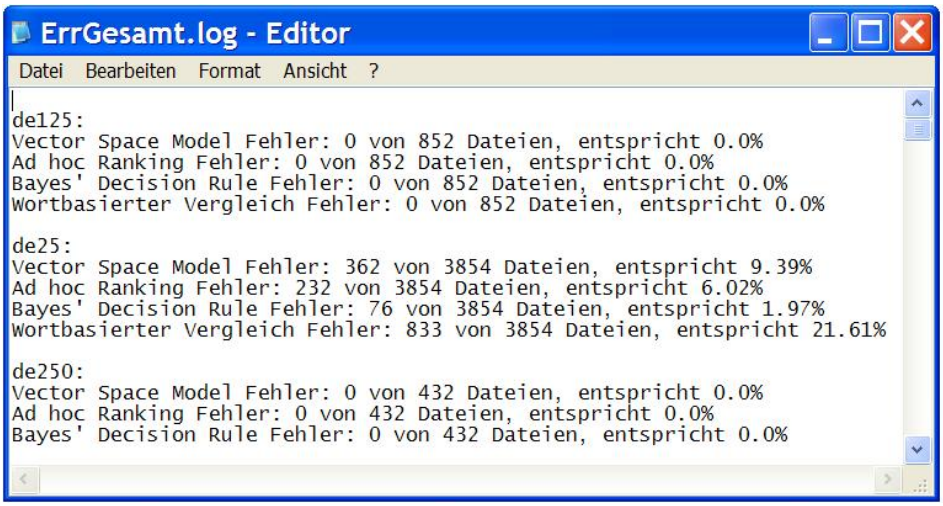


Abbildung 44: Auszug aus der ErrGesamt.log-Datei

Um festzustellen, ob das Testergebnis korrekt ist, wird der Name des Verzeichnisses mit dem Namen des Sprachmodells verglichen, dem die Datei zugeordnet wurde. Die Error-Dateien sind daher nur dann von Bedeutung, wenn

die Sprache des Dokuments bereits bekannt ist und der Name des Verzeichnisses den Namen des dazugehörigen Sprachmodells oder seine Abkürzung enthält. Andernfalls können die Testergebnisse nicht automatisch ausgewertet werden. Es ist außerdem zu beachten, dass in dem Fall, wenn der Testkorpus mit Hilfe des Werkzeuges SplitTest erstellt wurde, die angenommene und die tatsächliche Dokumentsprache nicht unbedingt identisch sein müssen. Wurde die Sprache von zwei oder mehr Methoden falsch identifiziert, liegt die Vermutung nah, dass der ursprüngliche Text Abschnitte in einer anderen Sprache enthalten hat. Ein 250 Byte langes Testdokument aus der ukrainischen Zeitung wurde zum Beispiel von drei Methoden dem Englischen zugeordnet, weil es ausschließlich Eigennamen in der englischen Sprache enthielt:
„Apax Partners, Ashmore, Baring Vostok, EFG-Hermes Private Equity, Actis, Templeton, Scudder, Montgomery, ChrysCapital, Delta“

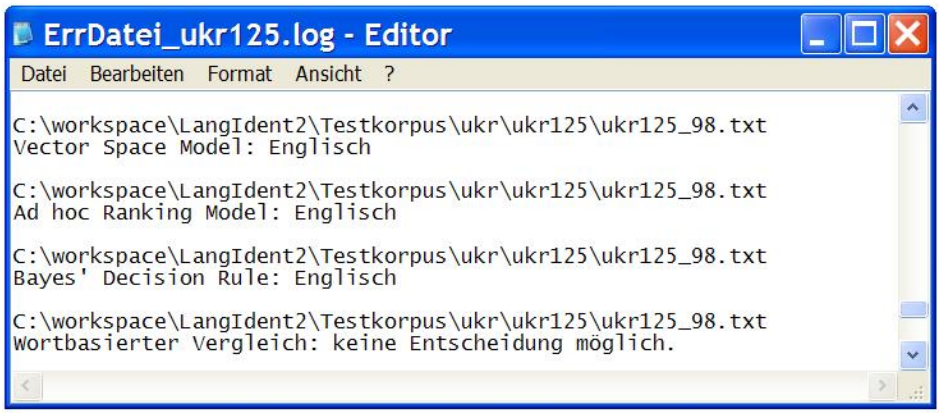


Abbildung 45: Auszug aus einer Error-Datei (2)

Mit Hilfe der Schaltfläche „Zurück zum Test“ oder des Menüs NAVIGATION/TEXTBEREICH ANZEIGEN (Tastenkürzel <Strg>+<T>) kann zu der Textansicht gewechselt werden.

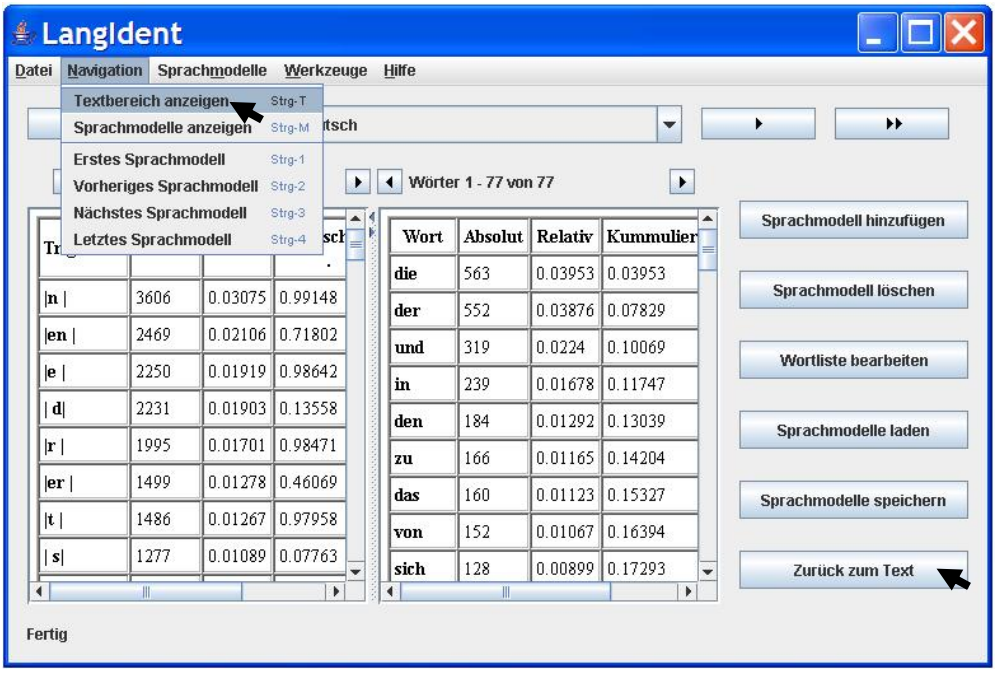


Abbildung 46: Benutzeroberfläche: Wechsel zur Textansicht

Über den Menüpunkt „Datei/Beenden“ oder mit dem Tastenkürzel <Alt>+<X> wird das Programm beendet.

13 Evaluierung

In diesem Kapitel wird die letzte Phase der LangIdent-Entwicklung beschrieben, in der die Systemperformanz bei der Sprachidentifikation von mono- und multilingualen Dokumenten getestet wurde.

13.1 Evaluierung des Systems bei der Sprachidentifikation von monolingualen Texten

Mit der Evaluierung des Systems bei der Sprachidentifikation von monolingualen Texten wurden zwei Hauptziele verfolgt. Zum einen sollten implementierte Methoden bzgl. deren Effektivität bei der Anwendung auf Testdokumente unterschiedlicher Länge miteinander verglichen werden. Zum anderen sollte herausgefunden werden, wie sich die Erweiterung des Systems um neue Sprachen auf seine Performanz auswirkt.
Die Evaluierung des Systems erfolgte in fünf Testdurchläufen. Im ersten Testlauf wurde die Systemperformanz bei der Identifikation von sechs Sprachen: Deutsch, Englisch, Spanisch, Französisch, Italienisch und Russisch überprüft. Nach der festgestellten Schwäche der wortbasierten Methode wurde eine Erweiterung der Methode vorgenommen (vgl. Punkt 9.2, S. 75), die im zweiten Test evaluiert wurde. In den folgenden zwei Testdurchläufen wurde das System zuerst um Tschechisch und dann um Ukrainisch erweitert.
Der letzte Test wurde auf einem neu erstellten Testkorpus, bestehend aus in denselben acht Sprachen verfassten Dokumenten, durchgeführt, um die Aussagekraft der in den vorherigen Tests erzielten Ergebnisse zu überprüfen.

13.1.1 Umfang und Aufbau von Testkorpora

Für die Evaluierung der Systemleistungsfähigkeit bei der Sprachidentifikation von monolingualen Texten wurden zwei Testkorpora aufgebaut, bestehend aus

Dokumenten in acht Sprachen: Deutsch, Englisch, Spanisch, Italienisch, Französisch, Russisch, Tschechisch und Ukrainisch. Dokumente für die Testkorpora wurden folgenden Online Zeitungen entnommen: „Spiegel Online“ (http://www.spiegel.de/), „Glasgow Herald“ (http://www.theherald.co.uk/), „Efe“ (http://www.efe.es/), „Le Monde“ (http://www.lemonde.fr/) „La Stampa Web“ (http://www.lastampa.it/redazione/default.asp), „Izvestia“ / „Известия“, (http://www.izvestia.ru/), „České Noviny.cz“ (http://www. ceskenoviny.cz), „Moje Noviny.cz“ (http://www.mojenoviny.cz), „Lidovsky Centrum.cz“ (http://lidovky. centrum.cz), „Day“ / „День“ (http://www.day.ua/), „ДзеркалоТижня“ (http://www.zn.kiev.ua).

Wie auch bei der Erstellung des Trainingskorpus wurden für jede der acht Sprachen ca. 200 KByte große Testdateien zusammengestellt. Jede große Datei wurde mit Hilfe des von den Autoren entwickelten Hilfsprogramms SplitText in kleinere Dateien der Länge von 25, 50, 125, 250 und 500 Zeichen geteilt, was jeweils 50, 100, 250, 500 und 1000 Byte entspricht, um die Performanz des Sprachidentifizierers an Dokumenten unterschiedlicher Länge testen zu können. Die Testdateien wurden nur an den Wortgrenzen geteilt, um eine unerwünschte Teilung in der Mitte eines Wortes zu vermeiden. Infolge dessen kann die Größe der erstellten Dateien um bis zu 50 Byte von den oben angegebenen Größen abweichen:

25 Zeichen: 52-116 Byte, 50 Zeichen: 102-166 Byte, 125 Zeichen: 252- 302 Byte, 250 Zeichen: 502- 554 Byte, 500 Zeichen: 1002-1044 Byte.

Die genauen Angaben zur Größe und Anzahl der Dokumente in beiden Testkorpora werden in den nachfolgenden Tabellen dargestellt.

Sprache	Anzahl der Testdokumente/ Größe (insg.)	Anzahl der Testdokumente von Länge …				
		50 Byte	100 Byte	250 Byte	500 Byte	1000 Byte
Deutsch	7023 / 203 KB	3657	1945	807	409	205
Englisch	7144 / 206 KB	3723	1975	821	416	209
Spanisch	7250 / 202 KB	3798	2001	825	417	209
Italienisch	7089 / 207 KB	3684	1962	818	416	209
Französisch	7152 / 204 KB	3748	1973	813	411	207
Russisch	6605/ 198 KB	3400	1827	781	397	200
Tschechisch	7126 / 204 KB	3722	1973	813	411	207
Ukrainisch	7091 / 204 KB	3704	1958	811	411	207

Tabelle 12: Anzahl der Testdokumente im ersten Testkorpus differenziert nach der Sprache und der Dokumentlänge

Sprache	Anzahl der Testdokumente/ Größe (insg.)	Anzahl der Testdokumente von Länge …				
		50 Byte	100 Byte	250 Byte	500 Byte	1000 Byte
Deutsch	7025 / 203 KB	3662	1943	807	408	205
Englisch	7504 / 216 KB	3911	2077	861	436	219
Spanisch	7237 / 206 KB	3800	1993	821	415	208
Italienisch	7155 / 209 KB	3712	1985	827	420	211
Französisch	7563 / 215 KB	3966	2084	861	434	218
Russisch	6865 / 207 KB	3521	1905	816	414	209
Tschechisch	7074 /202 KB	3708	1948	806	407	205
Ukrainisch	7046 / 204 KB	3668	1951	811	410	206

Tabelle 13: Anzahl der Testdokumente im zweiten Testkorpus differenziert nach der Sprache und der Dokumentlänge

13.1.2 Ergebnisse und Diskussion

Im ersten Testdurchlauf wurden alle vier Methoden auf einem Korpus getestet, der Dokumente in sechs Sprachen enthielt: Deutsch, Englisch, Spanisch, Französisch, Italienisch und Russisch. Dabei wurde die wortbasierte Methode in ihrer ursprünglichen Version eingesetzt (s. Punkt 9.2, S. 75). Die ermittelten durchschnittlichen Fehlerquoten werden in der unten abgebildeten Tabelle zusammengefasst.

Sprache	Dokument-Größe	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
Durchschnitt	50 Byte	*10,52%*	*7,29%*	*2,78%*	*24,82%*
	100 Byte	*3,48%*	*1,83%*	*0,42%*	*11,27%*
	250 Byte	*0,47%*	*0,08%*	*0,02%*	*4,36%*
	500 Byte	*0.04%*	*0%*	*0%*	*3,2%*
	1000 Byte	*0%*	*0%*	*0%*	*1,69%*

Tabelle 14: Durchschnittliche Fehlerquoten für sechs Sprachen differenziert nach der Klassifikationsmethode und der Dokumentgröße

Aus den in der Tabelle angegebenen Zahlen lässt sich erkennen, dass die beste Erkennungsqualität bei der Bayes'schen Entscheidungsregel erreicht wird, bereits bei den Texten von 50 Byte, was ca. 25 Zeichen entspricht, liegt hier die Fehlerquote unter 3%. Die übrigen Methoden zeigten sich bei der Sprachidentifikation kurzer Textabschnitte weniger erfolgreich, insbesondere betrifft das die wortbasierte Methode.

Bei allen vier Methoden konnte beobachtet werden, dass mit der wachsenden Testdokumentlänge erwartungsgemäß die Verbesserung der Erkennungsqualität stattfindet. Bei drei Methoden, Vector Space, Ad Hoc und Bayes'schen Entscheidungsregel, liegt die Fehlerrate bei der Spracherkennung von Texten mit der Länge von 250 Byte unter 1% und eine 100-prozentige Genauigkeit wird bei den Testdokumenten mit der Länge von 1000 Byte erreicht.

Die schlechtesten Ergebnisse werden bei der Sprachidentifikation mittels wortbasierter Methode erreicht. Die Differenzierung nach der Sprache ergibt, dass der größte Fehleranteil bei den verwandten Sprachen liegt, wie Französisch, Spanisch und Italienisch, wo die Dokumente in den meisten Fällen der falschen Sprache zugeordnet wurden, sowie bei stark flektierenden Sprachen wie Russisch, wo bei vielen Dokumenten keine Entscheidung möglich war.

In der nachfolgenden Graphik werden die nach Sprachen differenzierten Fehlerquoten der wortbasierten Methode in der abnehmender Reihenfolge dargestellt:

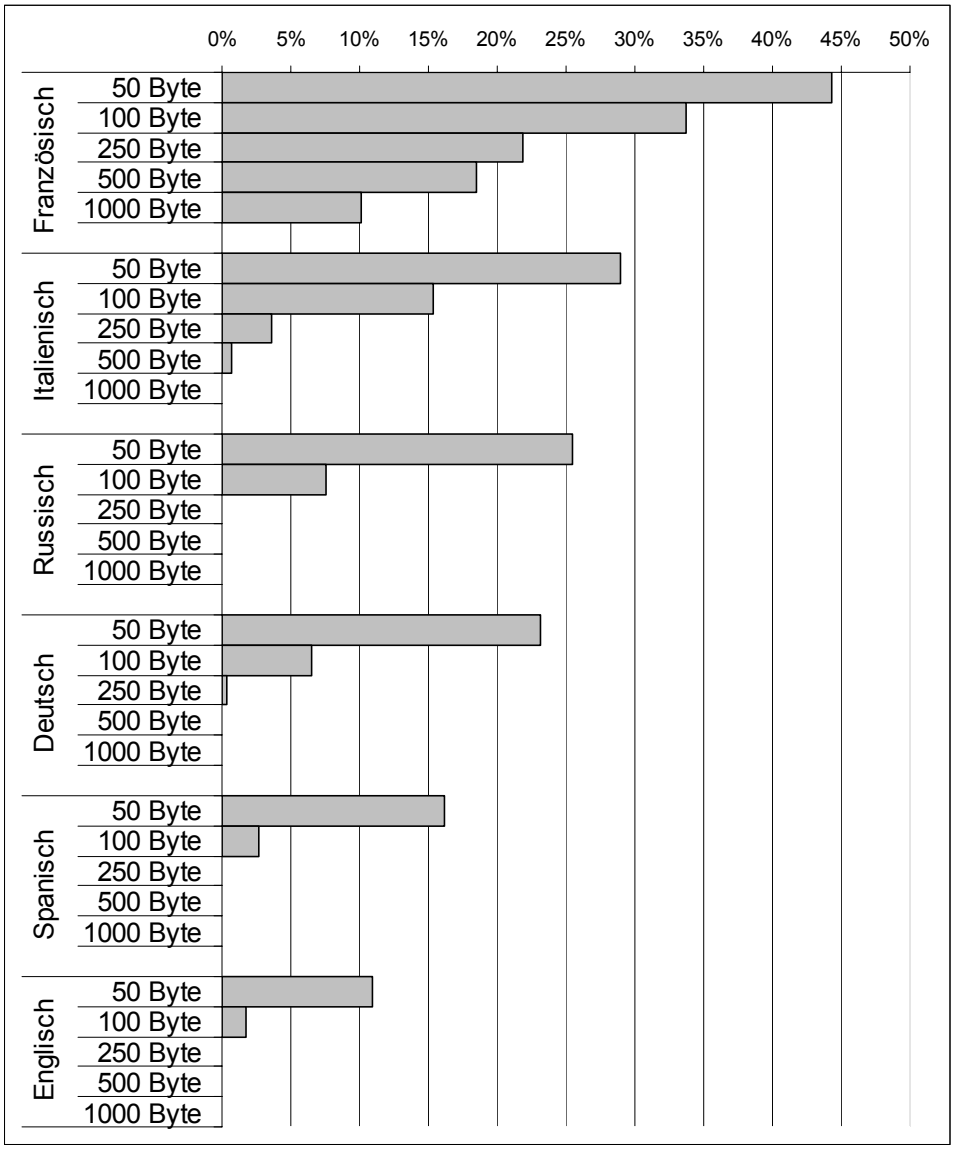


Abbildung 47: Wortbasierte Methode: Fehlerquoten differenziert nach der Sprache und der Dokumentgröße

Die Analyse von falsch klassifizierten französischen und italienischen Dokumenten ließ feststellen, dass viele dieser Texte als Spanisch identifiziert wurden.

Um den Grund für das auftretende Problem aufzudecken, wurden die Wortlisten dieser Sprachen miteinander verglichen. Es wurde herausgefunden, dass viele der in allen drei Sprachen häufig vorkommenden Kurzwörter, im spanischen Sprachmodell höhere relative Häufigkeiten aufweisen:

Kurz-wort	Spanisch		Französisch		Italienisch	
	Absolute Häufigkeit	Relative Häufigkeit	Absolute Häufigkeit	Relative Häufigkeit	Absolute Häufigkeit	Relative Häufigkeit
De	1227	0,07354	950	0,06077	-	-
La	725	0,04345	482	0,03083	358	0,02241
Que	583	0,03494	156	0,00998	-	-
En	482	0,02889	209	0,01337	-	-
A	328	0,01966	175	0,01119	237	0,01483
Del	237	0,01421	-	-	193	0,01208
Con	167	0,01001	-	-	128	0,00801

Tabelle 15: Spanische, französische und italienische Kurzwörter

Das heißt: wenn in einem französischen / italienischen Testdokument einzelne hoch frequente Wörter gefunden werden, die auch in der spanischen Wortliste enthalten sind und höhere Werte aufweisen, kann dieses Dokument in vielen Fällen dem Spanischen zugeordnet werden.

Der französische Text „**sur la** pour**suite de la** croi**ssance**“ wird zum Beispiel dem Spanischen zugeordnet, weil die Summe der relativen Häufigkeiten von den im spanischen Sprachmodell enthaltenen *la* und *de* den Wert von addierten relativen Häufigkeiten von *sur*, *la* und *de* aus der französischen Wortliste übertrifft.

Viele italienische Testdokumente werden ebenfalls als Spanisch identifiziert, zum Beispiel: „**per** confermare **la** fedelta'“; „Questa **sua** richiesta **la**“; „**del** Fronte Veneto Skinhead, fondato tre **anni** fa **a**“ u.v.m.

Das Problem betrifft insbesondere kurze Textabschnitte (50-100 Byte), weil es sehr wahrscheinlich ist, dass darin überwiegend häufige, in allen drei Sprachen vorkommende Kurzwörter enthalten sind.

Um der beschriebenen Problematik bei der Identifikation von eng verwandten Sprachen entgegen zu wirken, wurde die wortbasierte Methode modifiziert.

Nach der Verbesserung der Methode wurde ein neuer Test auf demselben Sprachkorpus durchgeführt. Die ermittelten Ergebnisse weisen eine deutlich verbesserte Genauigkeit der wortbasierten Methode auf, vor allem bei der Sprachidentifikation von französischen und italienischen Testdokumenten. Bei den übrigen Sprachen verändern sich die Erkennungsquoten unbedeutend: die Abweichung liegt unter 2%.

Zur Veranschaulichung der gewonnenen Ergebnisse dient die nachfolgende Tabelle, in der die Fehlerquoten differenziert nach der Sprache und der Dokumentgröße zusammengefasst werden:

Sprache	Dokument-größe	Wortbasierte Methode (ursprünglich)	Wortbasierte Methode (modifiziert)
Deutsch	50 Byte	*23,14%*	*21,59%*
	100 Byte	*6,52%*	*4,63%*
	250 Byte	*0,35%*	*0%*
	500 Byte	*0%*	*0%*
	1000 Byte	*0%*	*0%*
Englisch	50 Byte	*10,95%*	*11,01%*
	100 Byte	*1,75%*	*1,7%*
	250 Byte	*0,12%*	*0,12%*
	500 Byte	*0%*	*0%*
	1000 Byte	*0%*	*0%*
Spanisch	50 Byte	*16,14%*	*15,4%*
	100 Byte	*2,65%*	*3,1%*
	250 Byte	*0,12%*	*0,48%*
	500 Byte	*0%*	*0%*
	1000 Byte	*0%*	*0%*
Italienisch	50 Byte	*28,91%*	***22,99%***
	100 Byte	*15,39%*	***5,71%***
	250 Byte	*3,55%*	***0%***
	500 Byte	*0,72%*	*0%*
	1000 Byte	*0%*	*0%*
Französisch	50 Byte	*44,32%*	***28,9%***
	100 Byte	*33,76%*	***8,31%***
	250 Byte	*21,89%*	***0,62%***
	500 Byte	*18,49%*	***0%***
	1000 Byte	*10,14%*	***0%***
Russisch	50 Byte	*25,44%*	*25,44%*
	100 Byte	*7,57%*	*7,57%*
	250 Byte	*0,12%*	*0,12%*
	500 Byte	*0%*	*0%*
	1000 Byte	*0%*	*0%*

Tabelle 16: Wortbasierte Methode: Fehlerquoten differenziert nach der Sprache und der Dokumentgröße

Die angestrebte Verbesserung der Erkennungsgenauigkeit durch die Änderung der wortbasierten Methode kann am Beispiel des oben aufgeführten französischen Texts demonstriert werden:

Das im ersten Testdurchlauf falsch erkannte Dokument „**sur la** poursuite **de la** croissance“ wird in der neuen Testrunde korrekt identifiziert, weil die Anzahl der im französischen Sprachmodell gefundenen Wörter größer als die Anzahl der in der spanischen Liste gefundenen Wörter ist (drei im spanischen Sprachmodell vs. zwei im französischen).

Wenn die Prozentzahlen aus der Tabelle 16 betrachtet werden, fällt auf, dass trotz der erreichten Verbesserung der Anteil der falsch zugeordneten französischen und italienischen Testdokumente höher im Vergleich zum Spanischen ist. Die Untersuchung von Fehlerdateien zeigte, dass die unkorrekte Klassifizierung immer dann passiert, wenn die Anzahl der in der französischen / italienischen Wortliste gefundenen Wörter mit der Anzahl der Wörter aus der spanischen Liste übereinstimmt. In diesem Fall werden die relativen Häufigkeiten der gefundenen Wörter summiert, was letztendlich zu dem falschen Ergebnis führt.

Folgende französische Testdokumente wurden im zweitem Testdurchlauf fälschlicherweise dem Spanischen zugeordnet, weil in diesen kurzen Textabschnitten die einzigen Wörter *en*, *la*, *de*, *que*, nach denen die Klassifikation erfolgt, sowohl in der französischen als auch in der spanischen Wortliste vorkommen, jedoch im Spanischen größere Werte haben:

„résistant, toujours *en* butte“
„*la* réforme fiscale dénote“
„*la* première priorité *de*“
„*de la* direction *de la* FEN“
„l'attitude intransigeante *que* son parti avait adoptée,“

Das gleiche Problem betrifft auch folgende italienische Texte:

„presidente *del* Consiglio“
„*del* ministero *a* Firenze.“
„dai capelli grigi *con la*“
„*al* nazismo bisognerebbe portarli *a* fare *un* giro“

Die Auswertung der gewonnenen Ergebnisse sowohl im ersten, als auch im zweiten Testdurchlauf zeigt, dass sich die wortbasierte Methode im Vergleich zu den anderen Methoden durch eine größere Ungenauigkeit bei der Identifikation

kurzer Texte auszeichnet. Mit dieser Erkenntnis wird eine der in dem Punkt 2.6 angesprochenen Schwachstellen der wortbasierten Klassifikation bestätigt.

Die im Laufe der Tests ermittelten Fehlerraten der wortbasierten Methode setzten sich nicht nur aus der Anzahl der Texte zusammen, die fehlerhaft klassifiziert wurden, sondern auch aus den Dokumenten, bei denen keine Entscheidung getroffen werden konnte. Zu den letzteren gehören Dokumente, die keine Wörter oder Wortformen enthalten, anhand derer die Dokumentsprache identifiziert werden kann. Typischerweise enthalten solche Dokumente Aufzählungen, Eigennamen sowie unvollständige Sätze. Aufgrund der automatisierten Erstellung der Testkorpora konnte die Teilung des Textes inmitten eines Satzes nicht vermieden werden. Nachfolgend werden einige Beispiele solcher nicht-standardisierten Texte gegeben:

„Reisen vorbereitete, Reden“; „Rudolf Seiters, Manfred“ (Deutsch, 50 Byte)

„konnten. Josef Streit, ehemals Generalstaatsanwalt“; „Paragraph wirkungslos. Bonner Liberale kritisieren“ (Deutsch, 100 Byte)

„Cox, Menzieshill's international“; „shed 11,000 jobs last year“ (Englisch, 50 Byte)

„sophisticated street fighting -- however ''noble'“; „Singapore alleging passport forgery. Three officials“ (Englisch, 100 Byte)

„main supporting bout, newly-crowned British cruiserweight champion Dennis Andries meets experienced American journeyman Mike“ (Englisch, 250 Byte)

„предложения относительно сроков“; „сообщению, Козырев готов“ (Russisch, 50 Byte)

„мужчины. Проливные дожди нарушили работу городского“ (Russisch, 100 Byte)

„месяцами зарплаты, охваченные социальной паникой, обозленные всеми этими прелестями реформ", оставшиеся начинают бастовать.“ (Russisch, 250 Byte)

„Comerciales Internacionales, Liliana Canale, viajará,“; „Marco Agapito, Martín Hidalgo, Cesar Dulanto; Jorge“ (Spanisch, 100 Byte)

„Valenzuela). 0. Perú: Johnny Vega; Alfredo Ortiz, Jorge Delgado, Luis Guadalupe, Luis Altamirano; Víctor Morales, Marco“ (Spanisch, 250 Byte)

„mardi 8 février, entre“; „crédibilité : quarante-huit“ (Französisch, 50 Byte)

„(ADS), formation regroupant différentes strates“; „qu'une quelconque action militaire soit entreprise“ (Französisch, 100 Byte)

„Berlusconi ricevera' Clinton“; „bocciato. Sarebbe legittimo“ (Italienisch, 50 Byte)

„ben piu' numerosi. Semplicemente e' difficile riconoscerlo“; „contro petto. Qualche cittadino osserva dalle finestre.“ (Italienisch, 100 Byte)

Im dritten Testdurchlauf wurde das System um eine weitere Sprache - Tschechisch - erweitert, um zu beobachten, wie sich das auf die gesamte Systemperformanz auswirkt.

Das Hinzufügen von Tschechisch beeinflusste die Erkennungsqualität kaum: einige wenige festgestellte Abweichungen erreichen nicht einmal 2%:

Dok.-Größe	Sprachen	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
50 Byte	ohne Tschechisch	*10,52%*	*7,29%*	*2,78%*	*20,89%*
	mit Tschechisch	*10,21%*	*6,92%*	***3,19%***	***22,76%***
100 Byte	ohne Tschechisch	*3,48%*	*1,83%*	*0,42%*	*5,17%*
	mit Tschechisch	*3,41%*	*1,71%*	***0,6%***	***6,16%***
250 Byte	ohne Tschechisch	*0,47%*	*0.08%*	*0,02%*	*0,22%*
	mit Tschechisch	*0,46%*	*0,08%*	***0,05%***	***0,3%***
500 Byte	ohne Tschechisch	*0,04%*	*0%*	*0%*	*0%*
	mit Tschechisch	***0,07%***	*0%*	*0%*	***0,03%***
1000 Byte	ohne Tschechisch	*0%*	*0%*	*0%*	*0%*
	mit Tschechisch	*0%*	*0%*	*0%*	*0%*

Tabelle 17: Durchschnittliche Fehlerquoten für sieben Sprachen differenziert nach der Klassifikationsmethode und der Dokumentgröße

Die Differenzierung der in den drei Tests erhaltenen Ergebnisse nach einzelnen Sprachen zeigt, dass Russisch als die einzige kyrillische Sprache im Testkorpus die höchsten Erkennungsquoten hat. Bei allen Methoden, ausgenommen die wortbasierte, werden Testdokumente aller Längen mit der 100%-igen Genauigkeit erkannt.

Das änderte sich allerdings, als im nächsten Testdurchlauf die ukrainische Sprache hinzugefügt wurde. Die Identifikationsqualität beim Russischen verschlechterte sich deutlich. Besonders viele Fehler traten bei der Sprachidentifikation kurzer Testdokumente (50 Byte) auf. Dieses Problem lässt

sich auf den hohen Ähnlichkeitsgrad der beiden Sprachen sowie auf die geringe Anzahl von Wörtern bzw. Tri-Gramme in solchen Textabschnitten zurückführen.

Sprache	Dokument-Größe	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
Russisch ohne Ukrainisch	50 Byte	*0%*	*0%*	*0%*	*25,44%*
	100 Byte	*0%*	*0%*	*0%*	*7,57%*
	250 Byte	*0%*	*0%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Russisch mit Ukrainisch	50 Byte	***4,73%***	***2,47%***	***1,66%***	***30,58%***
	100 Byte	***1,59%***	***0,37%***	***0,16%***	***11,17%***
	250 Byte	***0,12%***	***0,12%***	*0%*	***0,74%***
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*

Tabelle 18: Vergleich von Fehlerquoten bei der Identifikation der russischen Sprache vor und nach dem Hinzufügen von Ukrainisch

Auf die gesamte Systemperformanz hatte das Hinzufügen des Ukrainischen keinen bedeutsamen Einfluss:

Dok.-Größe	Sprachen	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
50 Byte	ohne Ukrainisch	*10,21%*	*6,92%*	*3,19%*	*22,76%*
	mit Ukrainisch	***10,61%***	*6,91%*	***3,56%***	***24,53%***
100 Byte	ohne Ukrainisch	*3,41%*	*1,71%*	*0,6%*	*6,16%*
	mit Ukrainisch	***3,92%***	*1,63%*	***0,67%***	***7,33%***
250 Byte	ohne Ukrainisch	*0,46%*	*0,08%*	*0,05%*	*0,3%*
	mit Ukrainisch	***0,78%***	***0,09%***	*0,05%*	***0,49%***
500 Byte	ohne Ukrainisch	*0,07%*	*0%*	*0%*	*0,03%*
	mit Ukrainisch	***0,09%***	*0%*	*0%*	*0,03%*
1000 Byte	ohne Ukrainisch	*0%*	*0%*	*0%*	*0%*
	mit Ukrainisch	*0%*	*0%*	*0%*	*0%*

Tabelle 19: Durchschnittliche Fehlerquoten für acht Sprachen differenziert nach der Klassifikationsmethode und der Dokumentgröße

Um die Ergebnisse der ersten vier Testdurchläufe zu überprüfen, wurde ein zusätzlicher Test auf einem neu zusammengestellten Korpus durchgeführt, der Dokumente in denselben acht Sprachen enthielt. Die aus dem fünften Testdurchlauf gewonnenen Ergebnisse zeigen geringfügige Abweichungen von den Zahlen des vierten Tests (s. Anhang, S. 161).

In den nachfolgenden Graphik und Tabelle werden die durchschnittlichen Ergebnisse der beiden letzten Testdurchläufe zusammengefasst.
In der Graphik werden die Erkennungsquoten differenziert nach der Klassifikationsmethode und der Testdokumentlänge präsentiert:

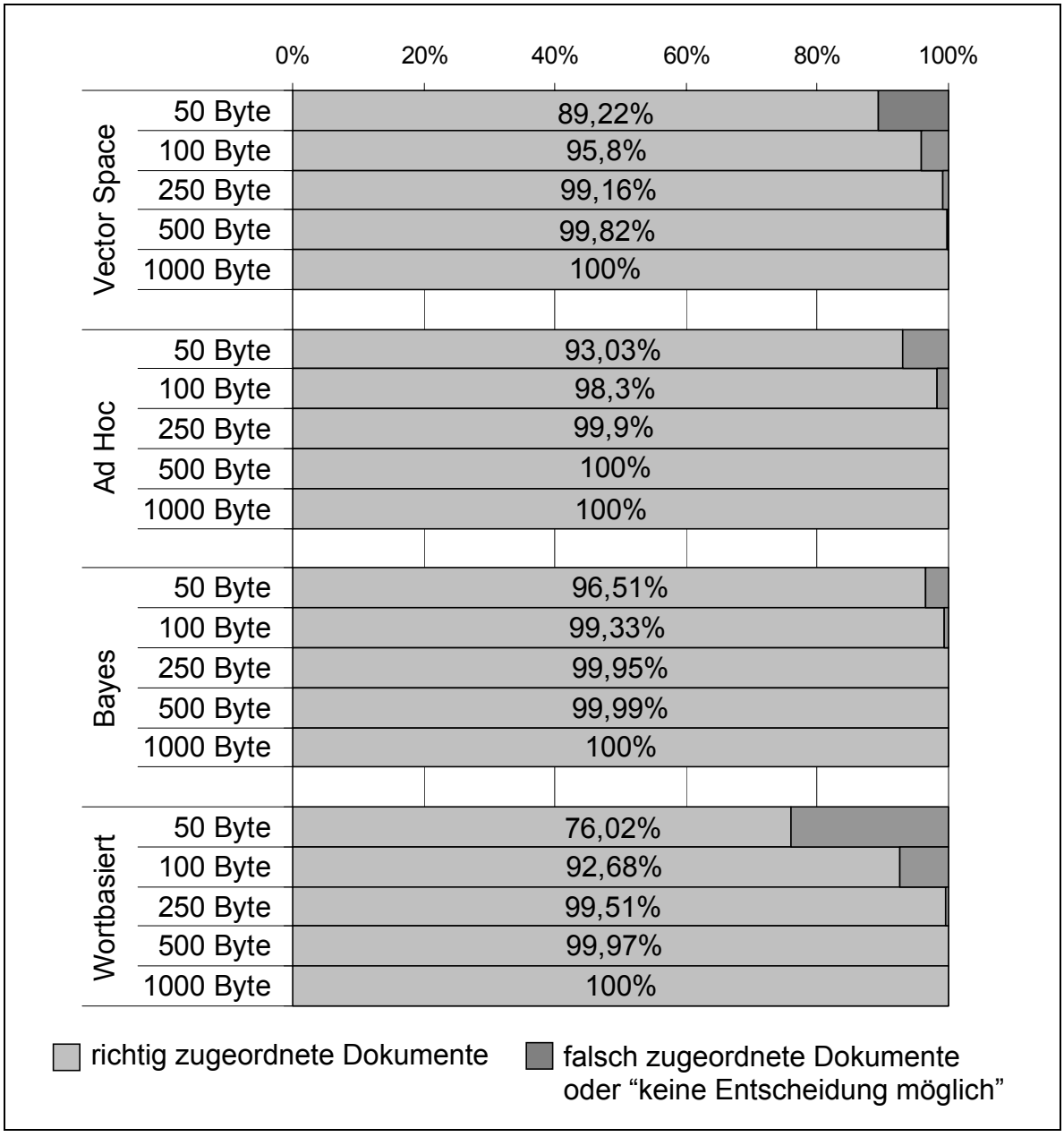


Abbildung 48: Durchschnittliche Erkennungsquoten für acht Sprachen differenziert nach der Klassifikationsmethode und der Dokumentgröße

In der Tabelle 20 werden die Fehlerquoten differenziert nach der Sprache, der Klassifikationsmethode und der Dokumentgröße dargestellt:

Sprache	Dokument-Größe	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
Deutsch	50 Byte	*8,84%*	*6,33%*	*2,14%*	*20,84%*
	100 Byte	*1,95%*	*1,56%*	*0,27%*	*4,55%*
	250 Byte	*0,06 %*	*0,06%*	*0%*	*0%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Englisch	50 Byte	*4,33%*	*6,05%*	*1,82%*	*12,88%*
	100 Byte	*0,65%*	*0,94%*	*0,12%*	*2,48%*
	250 Byte	*0%*	*0,06%*	*0,06 %*	*0,06%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Spanisch	50 Byte	*6,2%*	*10,63%*	*5,72%*	*16,12%*
	100 Byte	*1,35%*	*3,88%*	*0,92%*	*3,7%*
	250 Byte	*0%*	*0,24%*	*0%*	*0,42%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Italienisch	50 Byte	*19,97%*	*10,79%*	*2,12%*	*24,73%*
	100 Byte	*6,99%*	*2,64%*	*0,17%*	*6,58%*
	250 Byte	*0,73%*	*0,12%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Französisch	50 Byte	*28,26%*	*10,16%*	*4,12%*	*30,1%*
	100 Byte	*13,22%*	*1,97%*	*0,71%*	*9,34%*
	250 Byte	*2,92%*	*0,12%*	*0%*	*0,66%*
	500 Byte	*0,7%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Russisch	50 Byte	*5,25%*	*2,79%*	*1,91%*	*31,19%*
	100 Byte	*1,92%*	*0,31%*	*0,16%*	*11,86%*
	250 Byte	*0,36%*	*0,06 %*	*0%*	*1,1%*
	500 Byte	*0%*	*0%*	*0%*	*0,12%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Tschechisch	50 Byte	*4,2%*	*4,02%*	*4,79%*	*24,73%*
	100 Byte	*1,55%*	*0,99%*	*1,42%*	*7,71%*
	250 Byte	*0,31%*	*0,18%*	*0,25%*	*0,31%*
	500 Byte	*0%*	*0%*	*0%*	*0,12 %*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Ukrainisch	50 Byte	*9,21%*	*5,16%*	*5,32%*	*31,56%*
	100 Byte	*6,01%*	*1,33%*	*1,61%*	*12,38%*
	250 Byte	*2,41%*	*0%*	*0,06%*	*1,29%*
	500 Byte	*0,73%*	*0%*	*0,12%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*

Tabelle 20: Fehlerquoten bei der Identifikation von acht Sprachen differenziert nach der Sprache und der Dokumentgröße

Aus der Tabelle wird ersichtlich, dass sich die Bayes'sche Entscheidungsregel bei der Identifikation aller Sprachen außer Tschechisch und Ukrainisch als die beste Methode erwies. Bei der Erkennung dieser zwei Sprachen schnitt die Ad Hoc Ranking Methode etwas besser ab.

Der Trainingstext für Ukrainisch wurde im Unterschied zu den russischen Trainingsdaten nicht manuell bereinigt. Falls im ukrainischen Text lateinische Buchstaben vorkommen, hat es zur Folge, dass die Größe des Alphabets und damit auch die Übergangswahrscheinlichkeiten der Tri-Gramme falsch berechnet werden. Dies könnte die Erklärung für die etwas schlechteren Ergebnisse bei der Sprachidentifizierung mittels der Bayes'schen Entscheidungsregel bei Ukrainisch sein.

Die niedrigen Fehlerquoten bei der Erkennung des Tschechischen mit der Ad hoc Ranking Methode lassen vermuten, dass sich diese Methode besonders gut eignet, falls das Alphabet viele Sonderzeichen enthält. Durch den hohen Out-of-place Wert steigen die Distanzen für alle anderen Sprachen sehr schnell, und die Dokumentsprache kann eindeutig identifiziert werden.

Es muss an dieser Stelle darauf hingewiesen werden, dass die tatsächlichen Fehlerraten etwas niedriger sein durften als in der Tabelle 20 angegeben. Da der Trainingskorpus mittels SplitText erstellt wurde, konnte es vorkommen, dass die erzeugten Testdokumente einer Sprache einzelne Wörter oder sogar längere Textabschnitte in einer anderen Sprache enthalten (s. Kapitel 12, S. 117). Falls der fremdsprachige Textabschnitt ziemlich groß im Verhältnis zur Dokumentlänge war, wurde die Sprache dieses Abschnitts als die primäre Sprache des Testdokuments identifiziert, was durchaus korrekt war. Vom System wurde das Ergebnis jedoch als Fehler interpretiert.

Bei den kurzen Texten, bei denen die Fehlerquoten relativ hoch waren, konnten aus Zeitgründen nicht alle Testergebnisse überprüft werden, deshalb sind darin auch die oben beschriebenen „Fehler" enthalten, deren Eliminierung zur Senkung der in der Tabelle 20 angegebenen Fehlerquoten führen würde.

Im Folgenden werden die wichtigsten Ergebnisse bei der Sprachidentifikation von monolingualen Texten zusammengefasst:

1) Erwartungsgemäß erwies sich die wortbasierte Methode als uneffektiv bei der Sprachidentifikation kurzer Textabschnitte (50 - 100 Byte). Insbesondere betrifft das eng verwandte und stark flektierende Sprachen.
2) Die Bayes'sche Entscheidungsregel ist die Methode, bei der die besten Resultate erreicht werden.
3) Bei allen implementierten Methoden verbessern sich die Erkennungsraten mit der wachsenden Testdokumentlänge.
4) Die Erweiterung des Systems um neue Sprachen, die wenig mit den vorhandenen Sprachen korrelieren, nimmt wenig Einfluss auf die gesamte Systemperformanz. Betroffen werden in diesem Fall nur die Sprachen, die mit der hinzugefügten Sprache eng verwandt sind.

13.2 Evaluierung des Systems bei der Sprachidentifikation von multilingualen Texten

Die Evaluierung der Systemperformanz von LangIdent bei der Identifizierung mehrerer Sprachen in einem Dokument gestaltete sich etwas schwieriger als dies bei monolingualen Texten der Fall war. Zum einen, stand den Autoren kein multilingualer Testkorpus zur Verfügung, so dass zuerst nach geeigneten Dokumenten im Internet gesucht werden musste, was einige Zeit in Anspruch nahm. Eine Alternative wäre die automatisierte Erstellung des Testkorpus, wobei die monolingualen Dateien nach dem Zufallsprinzip zusammengefügt werden könnten. Aus Zeitgründen konnte diese Idee nicht umgesetzt werden. Stattdessen wurde ein Teil des multilingualen Testkorpus aufgebaut, indem die monolingualen Texte manuell zusammengefügt wurden.

Ein weiteres Problem stellte die Bewertung der Ergebnisse dar. Bei den monolingualen Texten war es relativ einfach, weil das einzige Bewertungskriterium war, ob die identifizierte Sprache mit der Sprache identisch ist, in der das Dokument tatsächlich verfasst ist. Bei der Bestimmung von mehreren Sprachen war die Entscheidung dagegen nicht immer eindeutig. Wurde beispielsweise in einem deutschen Textabschnitt ein Wort als Französisch identifiziert, konnte das korrekt sein, jedoch nicht das von dem

Benutzer gewünschte Systemverhalten darstellen. Die Entscheidung war dementsprechend nicht ganz frei von den subjektiven Wahrnehmungen des Testenden. Um eine weitgehende Objektivität bei der Bewertung zu erreichen, mussten die einzelnen Kriterien für die Bewertung genau formuliert werden:

1) Das System soll alle im Dokument enthaltenen Sprachen korrekt identifizieren. Die Erkennung von einer weiteren, nicht im Testdokument enthaltenen Sprache wird als Fehler betrachtet.

2) Das System soll die Position des Sprachwechsels im Text möglichst genau bestimmen. Die Abweichungen von eins bis vier Wörtern sind zulässig.

2.1 Umfang und Aufbau von Testkorpora

Um die Systemperformanz bei der Identifizierung multilingualer Dokumente zu testen, wurden zwei Testkorpora aufgebaut. Der erste Testkorpus enthält momentan 60 authentische multilinguale Dokumente aus dem Internet. Da LangIdent in der aktuellen Version keine HTML-Dateien unterstützt, wurden die Texte im *txt.*-Format mit der Codierung Unicode Big Endian abgespeichert. Manche Texte wurden außerdem gekürzt. Die Größe der Dateien variiert zwischen 256 Byte und 3170 Byte.

Der zweite Testkorpus enthält 100 Dateien, die aus Texten des monolingualen Testkorpus manuell zusammengesetzt wurden. Bei der Zusammensetzung der Dateien wurden keine zusätzlichen Satzzeichen oder Leerzeichen hinzugefügt. Ein typisches Beispiel einer auf diese Weise erstellten Datei stellt der folgende, in drei Sprachen verfasste Text dar:

begann, laut Scotland Yard, "die aufwendigste Detektivarbeit" with enormous competition. Ajax wanted him to replace de Salman Rushdie. Et deux attentats commis à Téhéran (Deutsch, Englisch, Französisch)

Einen Teil des künstlich aufgebauten Testkorpus bilden zweisprachige Dateien. Davon sind 50 vom Typ XY, wobei X und Y für zwei verschiedene Sprachen stehen:

verklagen den SPIEGEL auf einhundert Millionen. OUTÉ aurelèvement du loyer de l'argent décidé le (Deutsch, Französisch)

Die restlichen 20 Dateien wurden nach dem Muster XYX zusammengesetzt:

похоронные команды, выносящие раненых, шагают по, so ill at ease. There's no room for emotion in a трупам российских солдат, засыпанных обломками.(Russisch, Englisch, Russisch)

Außerdem enthält der Testkorpus 30 dreisprachige Texte. In der Tabelle unten werden multilinguale Testdokumente differenziert nach deren Anzahl und Größe zusammengefasst:

	Texte aus dem Internet	Künstlich erstellte Texte		
		Zweisprachig/ Typ XY	Zweisprachig/ Typ XYX	Dreisprachig
Anzahl der Dateien	62	50	20	31
Min. Dateigröße in Byte	256	106	300	256
Max. Dateigröße in Byte	3170	536	802	922
Durchschnittliche Größe in Byte	1172	263	548	538

Tabelle 21: Zusammensetzung der multilingualen Testkorpora

Die durchschnittliche Größe der selbst erstellten Dateien ist wesentlich kleiner als die Größe der Dateien aus dem Internet. Das System sollte vor allem an kurzen Texten getestet werden, da mit der wachsenden Dateigröße eher die Verbesserung der Performanz erwartet werden kann.

13.2.2 Ergebnisse und Diskussion

Die meisten Texte wurden mit der Einstellung der Intervalllänge von sechs Wörtern getestet. Nur dann, wenn die Texte besonders kurz waren, d. h. wenn die Textabschnitte in den einzelnen Sprachen nur drei bis fünf Wörter enthielten, wurde das Intervall auf drei oder vier gesetzt.

Identifikationsergebnisse der zweisprachigen Texte des Typs XY

Von insgesamt 50 zweisprachigen Texten des Typs XY wurden bei 48 Texten die beiden Sprachen korrekt erkannt. Die beiden Fehler traten bei der Identifizierung von Russisch und Ukrainisch auf. In einem Fall wurde ein ukrainischer Text dem Russischen zugeordnet, im zweiten wurde ein Teil des ukrainischen Textes als Russisch identifiziert. Bei der Analyse der falsch klassifizierten Dateien ließ sich feststellen, das zum einen die zu identifizierenden Strings zu kurz waren, zum anderen war die Ähnlichkeit des betreffenden Textes in den beiden Sprachen sehr hoch:

члена редколегії газети (der als Russisch erkannte ukrainische Text)
члена редколлегии газеты (der gleiche Text auf Russisch)
«Дня» політолога Вадима Карасьова.(der als Russisch erkannte ukrainische Text)
«Дня» политолога Вадима Карасева. (der gleiche Text auf Russisch)

Es ist zu erkennen, dass die beiden Texte nur anhand weniger Zeichen voneinander unterschieden werden können.

Die Position des Sprachwechsels wurde bei 19 Texten genau bestimmt, bei 20 Texten wurde sie um ein Wort nach links verschoben, bei acht Texten um zwei Wörter, und lediglich bei einem Text um drei Wörter.

Der Ähnlichkeitsgrad der im Text enthaltenen Sprachen scheint keinen direkten Zusammenhang mit der Genauigkeit bei der Bestimmung der Position des Sprachwechsels zu haben.

Bei den Texten, wo die Position des Sprachwechsels aufs Wort genau bestimmt wurde, sind die Kombinationen der eng verwandten Sprachen und der leicht zu unterscheidenden Sprachen zu ungefähr gleichen Teilen repräsentiert (neun vs. zwölf). Bei den Texten, in denen die Grenze um zwei Wörter verschoben wurde, überwiegen die Texte, die aus verwandten Sprachen zusammengesetzt wurden (fünf von sieben Texten). Auch scheint die Länge des zu identifizierenden Strings keinen Einfluss auf die Erkennung der Grenzen zu haben.

Identifikationsergebnisse der zweisprachigen Texte des Typs XYX

Bei den 19 der insgesamt 20 Texte wurden die beiden Sprachen korrekt erkannt. Bei einem Text wurde ein Teil des spanischen Textes als Italienisch identifiziert. Der betreffende Text enthielt eine Aufzählung von Eigennamen:

de Panamá, Costa Rica, Nicaragua,

Die zweisprachigen Texte aus dieser Kategorie enthalten zwei Sprachübergänge. Bei der Bestimmung der Abweichung von der richtigen Position des Sprachwechsels wurde die größere der beiden Abweichungen genommen. Dementsprechend sind die Ergebnisse etwas schlechter als bei den Texten mit nur einem Sprachübergang.
Bei vier Dateien wurde die Position der beiden Sprachwechsel aufs Wort genau erkannt. Bei elf Dateien wurde sie in einem oder in den beiden Fällen um ein Wort verschoben. In zwei Fällen betrug die Abweichung zwei Wörter, bei den restlichen zwei Dateien war sie gleich drei. Wieder konnte kein Zusammenhang zwischen dem Verwandtschaftsgrad der beiden im Text enthaltenen Sprachen und der Genauigkeit bei der Bestimmung der Position des Sprachwechsels beobachtet werden.

Identifikationsergebnisse der dreisprachigen Texte

Bei den 30 der 31 Texte wurden alle drei Sprachen korrekt erkannt. In einem Fall wurde ein Teil eines ukrainischen Textes als Russisch identifiziert:

ставимо». За словами О. (Ukrainisch, als Russisch erkannt)
ставим». За словами О. (Russisch)

Der Vergleich der beiden Strings macht deutlich, wie sehr sich die beiden Sprachen ähneln.
Die Position des Sprachwechsels wurde bei 12 Dateien genau festgestellt, bei 14 – um ein Wort verschoben, bei weiteren vier um zwei Wörter verschoben.

Identifikationsergebnisse der Texte aus dem Internet

Die meisten Texte waren zweisprachig, es gab darunter auch wenige dreisprachige Texte. Einen großen Anteil davon (27) machten parallele Texte aus, wo der gleiche Text zuerst in einer und danach in einer anderen Sprache verfasst war, wobei die einzelnen Teile Wörter in einer anderen Sprache enthalten können (Beispiele s. unten). 14 Texte enthielten Zitate oder Originaltitel von Büchern oder Filmen in einer anderen Sprache. Die restlichen Dateien enthielten unterschiedliche Texte in mehreren Sprachen.

Auch wenn die authentischen Texte im Durchschnitt etwas länger waren, als die selbst erstellten Dokumente, waren sie oft schwieriger zu analysieren. Dies soll an einigen Beispielen demonstriert werden, in denen in dem ersten in der deutschen Sprache verfassten Absatz eines parallelen Textes mehrere englische Wörter vorkommen:

Über den großen Teich hinaus

||| Oliver Weiss

Kommt Ihnen dieses Motiv bekannt vor? Der **Illustrator** dokumentiert **step by step**, wie er seine eigene Version des berühmten „**New Yorker**“-Posters mit **Photoshop** erstellt.

To Infinity and Beyond: Does this image look familiar? Here is a walkthrough of how the illustrator made his own version of the famous cover from “The New Yorker” magazine with Photoshop.

Shock in the Ear' wurde als Installation vom **New Media Arts Fund of the Australia Council** unterstützt und als CD-ROM von der australischen Filmkommission finanziert. Die CD-ROM wurde im Juni 1998 fertiggestellt.

Shock in the Ear was funded as an installation by the New Media Arts Fund of the Australia Council, and as a stand-alone CD-ROM by the Australian Film commission. The CD-ROM was completed in June, 1998.

Ein weiteres anschauliches Beispiel stellt der nächste Text dar, in dem englische und deutsche Wörter gemischt sind:

KEYWORDS: Portraits, Portrait, Porträt, Porträts, Portraitkarikaturen, Porträtkarikaturen, Karikatur, Karikaturen, politische Karikaturen, caricature, caricatures, Gesichter, faces, actors, Schauspieler, actor, Prominente, Promi, Promis, Berühmtheiten, celebrities, celebrity, celebs, celeb, Engel, angel, Wolke, cloud, Weihnachtsmann, Santa Claus, Weihnachten, Christmas, Xmas,

Kriss Kringle, Christkind, fliegen, Sterne, stars, Sänger, singer, singers

Bei 58 der 60 Dateien wurden alle im Dokument enthaltenen Sprachen korrekt erkannt. In vier Fällen wurde eine zusätzliche, nicht im Dokument enthaltene Sprache identifiziert (zwei Mal Italienisch, zwei Mal Französisch). Dass gerade die Sprachen der romanischen Familie erkannt werden, könnte damit zusammenhängen, dass die meisten Sprachen des lateinischen Alphabets viele Wörter und Stämme aus dem Lateinischen übernommen haben. Ist solch ein Wort in dem Textabschnitt enthalten, erhöht sich die Wahrscheinlichkeit für die Sprachen, die dem Lateinischen am ähnlichsten sind.

Eine mögliche Lösung für das Problem könnte die Angabe eines minimalen Anteils der Sprache an dem Gesamttext darstellen. Auf diese Weise würden die Sprachen unberücksichtigt bleiben, die weniger als 5% des Gesamttextes ausmachen. Selbst bei den relativ kurzen Texten liegt der Anteil der zusätzlichen Sprachen in den meisten Fällen bei weniger als 8%.

Die Erhöhung des Intervalls kann ebenfalls dazu führen, dass die Sprachen korrekt identifiziert werden, nur können dann eventuell nicht einzelne Wörter erkannt werden. Auch die Genauigkeit bei der Bestimmung der Position des Sprachwechsels wird dadurch verschlechtert.

Die Position des Sprachwechsels wurde bei 18 Dateien korrekt bestimmt, bei weiteren 16 wurde sie um ein Wort verschoben, bei 16 – um zwei Wörter, bei sechs um drei Wörter und nur in zwei Testdokumenten wurde die Position des Sprachwechsels um vier Wörter verschoben.

Die Analyse der im Testkorpus enthaltenen Dateien hat die Vermutung bestätigt, dass der Sprachwechsel in der Regel durch Einführungszeichen oder Klammern und am häufigsten durch einen Zeilenumbruch signalisiert wird:

MONTHLY statistics are published on this page. The statistics include a list of countries from which users have visited this site during the month, and the number of pages visited during the month. Statistics from past months are also available.

AUF dieser Seite werden monatliche Statistiken veröffentlicht. Die Statistiken beinhalten eine Liste von Ländern, aus denen Benutzer die Web-Seite im Laufe des Monats besucht haben und die Anzahl von Seiten, die im Laufe des Monats besucht wurden. Statistiken aus vergangenen Monaten sind ebenso erhältlich.

Mit Hilfe der syntaktischen Analyse konnte die Grenze der beiden Sprachen in den unten angeführten Beispielstexten genau auf das Wort bestimmt werden:

England, Irland, Frankreich, Spanien, Dänemark, Belgien und Deutschland waren die Stationen unserer Dreharbeiten zur Clipserie **»Learning for anyone, at any time, at any place«.** Unternehmen, Forscher und Anwender berichten in kurzen, 1-2minütigen Clips über ihre Erfahrungen mit technologie-unterstütztem Lernen,ihre Erwartungen und ihre Kritik.

Culture Shock – from Nairobi to Copenhagen: seeing Christianity from the other side **(Kulturschock – von Nairobi nach Kopenhagen: Christentum von der anderen Seite gesehen) hat den zweiten Preis in der Kategorie Dokumentarfilm beim 15. Europäischen TV Festival für religiöse Programme gewonnen, dem bedeutendsten Festival seiner Art.**

Die oben beschriebenen Ergebnisse, die bei der Sprachidentifikation von multilingualen Dokumenten erzielt wurden, können in der nachfolgenden Tabelle zusammengefasst werden.

Anzahl der Dateien		Korrekt erkannte Sprachen und die Position des Sprachwechsels… / verschoben um…				
		genau erkannt	1Wort	2 Wörter	3 Wörter	4 Wörter
Zweisprachig Typ XY	50	38%	40%	16%	2%	-
Zweisprachig Typ XYX	20	20%	55%	10%	10%	-
Dreisprachig	31	38,71%	45,16%	12,9%	-	-
Texte aus dem Internet	62	29,03%	25,81%	25,81%	9,68%	3,22%
Gesamt		95,33%				

Tabelle 22: Erkennungsquoten bei der Sprachidentifikation multilingualer Testdokumente

Die Ergebnisse können wie folgt zusammengefasst werden:

1) Die Leistungsfähigkeit von LangIdent bei der Sprachidentifikation von multilingualen Dokumenten kann aus der Sicht der Autoren als gut bewertet werden: in 95,33% aller Testdokumente (in 155 Testdokumenten aus 163) wurden die darin enthaltenen Sprachen korrekt erkannt.

2) Die Position des Sprachwechsels wird bei der Mehrheit der zu identifizierenden Dokumente genau oder nur um ein Wort verschoben bestimmt: z.B. in 38 % zweisprachiger Dokumente vom Typ XY wurde die Position des Sprachwechsels aufs Wort genau erkannt, in weiteren 40% um ein einziges Wort verschoben.
3) Die meisten Schwierigkeiten traten bei der Identifikation eng verwandter Sprachen und in den Fällen auf, wo es viele Sprachübergänge gab. Bei der Bewertung der Systemeffektivität muss in Betracht gezogen werden, dass in einigen Fällen, wo die Sprachen stark gemischt waren oder eng verwandte Sprachen in einem Text vorkamen, die Aufgabe der korrekten Sprachidentifikation sich sogar für einen Experten nicht einfach gestaltete.

Zusammenfassung und Ausblick

Die vorliegende Arbeit beschäftigt sich mit dem Problem der automatischen Sprachidentifikation von mono- und multilingualen elektronischen Textdokumenten. Diesem Thema wurden bereits viele wissenschaftliche Arbeiten gewidmet, in denen unterschiedliche Sprachmodellierungs- und Klassifikationsverfahren vorgestellt wurden. Allerdings gibt es nur wenige Arbeiten, die sich mit der Sprachidentifikation multilingualer Texte befasst haben. Im Internet sind keine Sprachidentifikationstools verfügbar, die über die Sprachidentifikation monolingualer Dokumente hinausgehen.

Im Gegensatz dazu identifiziert das im Rahmen der vorliegenden Arbeit entwickelte Sprachidentifikationssystem LangIdent sowohl die primäre Sprache als auch alle Sprachen, in denen ein elektronisches Testdokument verfasst ist. Für die Sprachmodellbildung werden im System der Tri-Gramm basierte und der wortbasierte Ansätze kombiniert: jedes Modell besteht aus einer Tri-Gramm- und einer Wortliste, wobei die Listenelemente jeweils mit einem Häufigkeitswert versehen sind. LangIdent ermöglicht eine aktive Mitwirkung der Benutzer bei der Sprachmodellierung. Die Klassifikation von monolingualen Dokumenten erfolgt unter Anwendung von einer der vier verfügbaren Methoden: Ad Hoc Ranking, Vektorraum Modell, Baeyes'schen Entscheidungsregel und wortbasierten Methode. Die Klassifikation von multilingualen Dokumenten erfolgt durch eine neu entwickelte Methode, die auf der Baeyes'schen Entscheidungsregel basiert.

Die Evaluierung von LangIdent auf einem umfangreichen Testkorpus, der Dokumente in acht Sprachen enthielt, erbrachte gute Ergebnisse. Bei der Sprachidentifikation von monolingualen Dokumenten erwies sich die Baeyes'sche Entscheidungsregel als die beste Methode für kurze Textdokumente. Bei den Dokumentenlängen von über 125 Zeichen betrugen die Erkennungsquoten mehr als 99% bei allen Methoden.

Auch bei der Sprachidentifikation von multilingualen Testdokumenten wurden gute Ergebnisse erzielt. Bei 95,33% aller mehrsprachigen Texte wurden die darin enthaltenen Sprachen korrekt erkannt. Die Position des Sprachwechsels wurde bei der Mehrheit der Dokumente genau oder nur um ein Wort verschoben bestimmt.

Auch wenn die Arbeit im Großen und Ganzen als gelungen betrachtet werden kann, gibt es sicherlich einige Stellen, an denen das Sprachidentifizierungssystem LangIdent noch weiter verbessert werden könnte.

In der aktuellen Version beschränkt sich das Programm auf eine einzige Kodierung - Unicode Big Endian. Trotz sorgfältiger Recherchen konnte keine Möglichkeit gefunden werden, die Kodierung eines Texts zuverlässig zu bestimmen. Es kann zwar anhand der sogenannten BOM (Byte Ordering Mark) festgestellt werden, ob das vorliegende Dokument in Unicode Big Endian oder Unicode Little Endian gespeichert ist, wenn diese Marke fehlt, kann nur geraten werden. Die weit verbreitete Methode, auf die lokale Einstellungen des Betriebssystems zuzugreifen, kann in vielen Fällen zwar ausreichend sein, ist bei einem Sprachidentifizierer allerdings nicht wirklich nützlich.

Das System könnte um ein Modul zur Kodierungserkennung erweitert werden, das anhand der Häufigkeitsverteilung verschiedener Zeichen und Steuersequenzen die Kodierung des Texts bestimmt. Bei der Spracherkennung würden Sprachen berücksichtigt werden, die in der erkannten Kodierung dargestellt werden können. Allerdings sollte es dem Benutzer möglich sein, die evtl. falsch erkannte Kodierung manuell zu berichtigen, um Folgefehler zu vermeiden.

Das Programm könnte auch so erweitert werden, dass es mit verschiedenen Dateiformaten umgehen kann, denn in der aktuellen Version wird nur das TXT-Format unterstützt. Es würde den Nutzen des Programms enorm steigern, da die Sprachidentifizierer häufig im Internet eingesetzt werden, wo sie mit HTML und XML – Dateien arbeiten. Zusätzlich würde es die Genauigkeit bei der Erkennung der Position der Sprachwechsels wahrscheinlich verbessern, da mit Formatierungen wie bold, oder kursiv, wie sie in HTML und RTF Dokumenten oft vorkommen, weitere Indizien für einen Sprachwechsel erhalten werden könnten.

Das Sprachidentifizierungssystem LangIdent wurde zwar so konzipiert, dass es um neue Sprachen erweitert werden kann, das gilt aber nur für die Sprachen, in denen es klare Wortgrenzen gibt. Die zurzeit unterstützten Sprachen gehören alle zu der europäischen Familie. Es wäre möglich, das System so zu erweitern, das auch die Sprachen unterstützt werden, in denen die Wortgrenzen nicht anhand von Leerzeichen ermittelt werden können (wie z.B. Japanisch oder

Chinesisch), oder die Sprachen, in denen die Schrift von rechts nach links verläuft (wie z.B. Arabisch).

Ein weiteres Tätigkeitsfeld wäre die Evaluierung des Programms. Aufgrund der großen Variierungsmöglichkeiten bei der Sprachmodellerstellung konnte die optimale Länge der Tri-Gramm- bzw. Wortliste nicht ermittelt werden. Auch wäre es interessant, zu untersuchen, wie die Qualität der Trainingsdaten die Erkennungsraten beeinflusst, ob manuell bereinigte Trainingstexte zu besseren Ergebnissen im Vergleich zu den nicht bearbeiteten Texten führt. Bei der Erstellung von monolingualen Testkorpora wurden von den Autoren Zeitungsartikel benutzt. Es wäre interessant, die Systemperformanz an weniger formellen Texten zu messen, wie z. B. Forenbeiträge.

Leider war es nicht möglich, in der für die Bearbeitung des Themas vorgesehenen Zeit einen umfangreichen multilingualen Testkorpus aufzubauen. Möglich wäre es, so einen Korpus automatisch zu erstellen, wobei die bestehenden monolingualen Texte nach Zufallsprinzip zu multilingualen Texten zusammengeknüpft werden. Auf diese Weise könnten Dokumente in verschiedensten Sprachenzusammenstellungen erstellt und getestet werden.

Literaturverzeichnis

ADAMS, Gary; RESNIK, Philip (1997):
"A Language Identification Application Built on the Java Client/Server Platform". In: BURSTEIN, Jill; Claudia LEACOCK (Hrsg.) (1997): *From Research to Commercial Applications: Making {NLP} Work in Practice*. Association for Computational Linguistics, Somerset, New Jersey.
(Zugriff: 14.03.2005, 15:34 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/738/http:zSzzSzumiacs.umd.eduzSz~resnikzSzpubszSzlanguage_idzSzadams_resnik.pdf/a-language-identification-application.pdf >

BASTRUP, Sofia; Christian PÖPPER (2003):
"Language Detection based on Unigram Analysis and Decision Trees".
(Zugriff: 14.03.2005, 16:20 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/30617/http:zSzzSzwww.cs.lth.sezSzEducationzSzCourseszSzEDA171zSzReportszSz2003zSzsofia_christina.pdf/bastrup03language.pdf >

BEESLEY, Kenneth R. (1998):
"Language Identifier: A Computer Program for Automatic Natural-Language Identification of On-line Text". In: *Languages at Crossroads: Proceedings of the 29th Annual Conference of the American Translators Association*.
(Zugriff: 6.05.2005, 15:40 MEZ)
<http://www.xrce.xerox.com/competencies/contentanalysis/tools/publis/langid.pdf >

CAPSTIK, Joanne et al. (1999):
„A System for Supporting Cross-Lingual Information Retrieval". In: *Information Processing and Management - Special topic issue „Web Research and Information Retrieval"*, Vol. 36 (1999), Nr. 2.
(Zugriff: 9.11.2004, 16:19 MEZ)
<http://www.coli.uni-sb.de/publikationen/softcopies/Capstick:1999:SSC.pdf>

CARSTENSEN, Kai-Uwe et al. (Hrsg.) (2004):
Computerlinguistik und Sprachtechnologie. Eine Einführung. 2., überarb. und erw. Aufl., München: Spektrum.

CAVNAR, William B.; John M. TRENKLE (1994):
"N-Gram-Based Text Categorization". In: *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, USA.
(Zugriff: 16.09.2004, 13:50 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/810/http:zSzzSzwww.info.unicaen.frzSz~giguetzSzclassifzSzcavnar_trenkle_ngram.pdf/n-gram-based-text.pdf >

CONSTABLE, Peter; Gary SIMONS (2000):
"Language identification and IT. Adressing problems of linguistic diversity on a global scale". In: *17th International Unicode Conference*, San José, California.(Zugriff: 2.11.2004, 16:45 MEZ)
<http://www.sil.org/silewp/2000/001/SILEWP2000-001.pdf>

COWIE, Jim; Yevgeny LUDOVIK; Ron ZACHARSKI (1999):
"Language recognition for mono- and multilingual documents". In: *Proceedings of the Vextal Conference*, Venice, November 22-24.
(Zugriff: 29.09.2004, 17:20 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/11087/http:zSzzSzclr.nmsu.eduzSz~razzSzpaperszSzVextalFinal.pdf/unknown.pdf >

DAMASHEK, Marc (1995):
"Gauging Similarity with n-Grams: Language-Independent Categorization of Text". In: *Science*. Vol. 267 (1995), 10. February, S. 843-848.

DAMASHEK, Marc (1995a):
Method of retrieving documents that concern the same topic, May 1995, U.S. Patent No. 5418951.

DUNNING, Ted (1994):
"Statistical Identification of Language". Technical Report MCCS 94-273, New Mexico State University, 1994.
(Zugriff: 16.09.2004, 14:00 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/36/http:zSzzSzwww.comp.lancs.ac.ukzSzcomputingzSzresearchzSzucrelzSzpaperszSzlingdet.pdf/dunning94statistical.pdf >

ELWORTHY, David (1998):
"Language Identification With Confidence Limits". In: *Proceedings of the Sixth Workshop on Very Large Corpora (COLING-ACL 98)*, S.94-101.
(Zugriff: 16.09.2004, 13:10 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/27686/http:zSzzSzacl.ldc.upenn.eduzSzWzSzW98zSzW98-1111.pdf/language-identification-with-confidence.pdf>

FRÖTSCHL, Bernhard; Wolf LINDSTROT (2004):
„Wahrscheinlichkeitstheorie und Hidden-Markov-Modelle". In: CARSTENSEN, Kai-Uwe et al. (Hrsg.) (2004): *Computerlinguistik und Sprachtechnologie. Eine Einführung*. 2., überarb. und erw. Aufl., München: Spektrum, S. 111-137.

GANESAN, Ravi; Alan SHERMAN (1993):
"Statistical Techniques for Language Recognition: An Introduction and Guide for Cryptanalysts. In: *Cryptologia*. Vol.17 (1993), Nr. 4.
(Zugriff: 29.09.2004, 17:24 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/1170/http:zSzzSzanubis.cs.umbc.eduzSzpubzSzREPORTSzSzcs-93-02.pdf/ravi93statistical.pdf >

GREFENSTETTE, Gregory (1995):
"Comparing two language identification schemes". In: *JADT 1995, 3rd International conference on Statistical Analysis of Textual Data*, Rome.
(Zugriff 9.05.2005, 14:00 MEZ)
<http://www.xrce.xerox.com/Publications/Attachments/1995-012/Gref---Comparing-two-language-identification-schemes.pdf>

KIKUI, Genichiro; Yoshihiko HAYASHI; Seiji SUZAKI (1996a):
"Cross-lingual Information Retrieval on the WWW". In: *Multilinguality in Software Engineering: The AI Contribution. European Coordinating Committee for Artificial Intelligence*, August 1996.
(Zugriff: 29.05.2005, 16:29)
<http://citeseer.ist.psu.edu/cache/papers/cs/22951/http:zSzzSzwww.slt.atr.co.jpzSz~gkikuizSzpaperszSz9608KikuiMULSAIC.pdf/kikui96crosslingual.pdf >

KIKUI, Gen-itiro (1996b):
"Identifying the Coding System and Language of On-line Documents on the Internet". In: *Proceedings of the 16th Conference on Computational linguistics*. Denmark, Vol. 2 (1996), S. 652-657.
(Zugriff: 29.05.2005, 16:31 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/22951/http:zSzzSzwww.slt.atr.co.jpzSz~gkikuizSzpaperszSz9608KikuiCOLING.pdf/kikui96identifying.pdf >

KIRCHHOFF, Katrin; Sonia PARANDEKAR; Jeff BILMES (2002):
„Mixed-memory Markov models for automatic language identification". In: *IEEE International Conference on Acoustics, Speech, and Signal Processing*, June 2002, Orlando Florida.
(Zugriff: 28.05.2005, 14:45 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/29495/http:zSzzSzssli.ee.washington.eduzSzpeoplezSzbilmeszSzmypaperszSzkatrin_icassp02.pdf/mixed-memory-markov-models.pdf >

KRENN, Brigitte; Christer SAMUELSSON (1997):
"The Linguists Guide to Statistics".
(Zugriff: 10.12.2004, 12:05 MEZ)
<http://www.ling.gu.se/~leifg/stat01/doc/dontpanic.ps>

LANGER, Stefan (2001):
„Sprachen auf dem WWW". In: LOBIN, Henning (Hrsg.): *Proceedings der GLDV-Frühjahrstagung 2001*, Universität Gießen, S. 85-91.
(Zugriff: 9.11.2004, 16:12 MEZ)
<http://www.uni-giessen.de/germanistik/ascl/gldv2001/ proceedings/>

LANGER, Stefan (2002):
„Grenzen der Sprachenidentifizierung".
(Zugriff: 14.11.2004, 21:00 MEZ) <http://konvens2002.dfki.de/cd/pdf/19V-langer.pdf>

LINS, Rafael D.; Paulo GONÇALVES (2004):
„Automatic Language Identification of Written Texts“. In: *ACM symposium on Applied Computing*, March 14-17, Nicosia, Cyprus, S. 1128-1133.

MARTINO, Michael J.; Robert C. PAULSEN (1996):
Language identification process using coded language words, August 1996, U.S. Patent No. 5548507.

MARTINO, Michael J.; Robert C. PAULSEN (1999):
Determining a natural language shift in a computer document, June 1999, U.S. Patent No. 5913185.

MARTINO, Michael J.; Robert C. PAULSEN (2001):
Natural language determination using partial words, April 2001, U.S. Patent No. 6216102 B1.

MARTINS, Bruno; Mário J. SILVA (2005):
“Language Identification in Web Pages”. In: *ACM Symposium on Applied Computing*, March 13-17, Santa Fe, New Mexico, USA, S. 764-768.

MUSTONEN, Seppo (1965):
„Multiple diskriminant analysis in linguistic problems“. In: *Statistical Methods in Linguistics*. Nr. 4 (1965), S. 37-44.

PADRÓ, Muntsa; Lluís PADRÓ (2004):
“Comparing methods for language identification”.
(Zugriff 26.05.2005, 18:26 MEZ)
<http://www.lsi.upc.es/~mpadro/publicacions/ idioma_sepln.pdf>

PETERS, Carol; Páraic SHERIDAN (2001):
“Multilingual Information Access”. In: AGOSTI, Maristella et al. (Hrsg.) (2001): *Lectures on Information Retrieval: Third European Summer-School*, *ESSIR 2000 Varenna, Italy*. Berlin, Heidelberg: Springer, S. 51-80.

PIOTROWSKI, Michael (1997):
“Statistical Language Identification for NLP-Supported Full Text Retrieval”, Master’s Thesis – Status Report 5.
(Zugriff: 29.05.2005, 16:38 MEZ)
<http://www.dynalabs.de/mxp/magister/ma_report_5.pdf>

POUTSMA, Arjen (2001):
“Applying Monte Carlo Techniques to Language Identification”. In: *CLIN 2001, Twelfth Meeting of Computational Linguistics in the Netherlands*, University of Twente, Enschede.
(Zugriff: 29.05.2005, 16:42 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/32415/http:zSzzSzwww.xs4all.nlzSz~ajwpzSzlangident.pdf/applying-monte-carlo-techniques.pdf >

PRAGER, John M. (1999):
"Linguini: Language Identification for Multilingual Documents". In: *Proceedings of the 32nd Hawaii International Confrence on System Sciences*, 1999.
(Zugriff: 29.09.2004, 18:17 MEZ)
<http://csdl.computer.org/comp/proceedings/hicss/1999/0001/02/00012035.PDF>

SALTON, Gerard; Michael J. McGILL (1987):
Information Retrieval – Grundlegendes für Informationswissenschaftler / Wolfgang von KEITZ (Übers.) Hamburg, New-York: McGraw-Hill.

SCHMITT, John C. (1991):
Trigram-based method of language identification, October 1991, U.S. Patent No. 5062143.

SCHULZE, Bruno M. (2000):
Automatic language identification using both N-Gram and word information, December 2000, U.S. Patent No. 6167369.

SIBUN, Penelope; Lawrence A. SPITZ (1994):
"Language Determination: Natural Language Processing from Scanned Document Images". In: *Proceedings of the Fourth {ACL} Conference on Applied Natural Language Processing*, 13-15 October 1994, Stuttgart, S. 15-21.
(Zugriff:16.09.2004, 13:32 MEZ)
<http://portal.acm.org/citation.cfm?id=974363&coll=ACM&dl=ACM&CFID=31149134&CFTOKEN=15172519>

SIBUN, Penelope; Jeffrey C. REYNAR (1996):
"Language Identification: Examining the Issues". In: *5th Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, USA.
(Zugriff: 16.09.2004, 13:20 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/810/http:zSzzSzwww.cis.upenn.eduzSz~jcreynarzSzsdair96.pdf/sibun96language.pdf >

SOUTER, Clive et al. (1994):
"Natural Language Identification using Corpus-Based Models". In: *Hermes Journal of Linguistics*. Vol. 13 (1994), S. 183-203.
(Zugriff: 12.11.2004, 12:18 MEZ)
<http://hermes2.asb.dk/archive/FreeH/H13_15.pdf>

SUZUKI, Izumi et al. (2002):
"A Language and Character Set Determination Method Based on N-gram Statistics". In: *ACM Transactions on Asian Language Information Processing*, Vol. 1 (2002), Nr. 3, S. 269-278.
(Zugriff: 29.05.2005, 16:45 MEZ)
<http://portal.acm.org/citation.cfm?id=772759&coll=ACM&dl=ACM&CFID=31149134&CFTOKEN=15172519>

TAN, Chew Lim; Peck Yoke LEONG; Shoujie HE (1999):
"Language Identification in Multilingual Documents". In: *International Symposium on Intelligent Multimedia and Distance Education*.
(Zugriff: 22.09.2004, 21:44 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/17585/http:zSzzSzwww.comp.nus.edu.sgzSzlabszSzchimezSzpaperdownloadzSzmultiling.pdf/language-identification-in-multilingual.pdf >

TORRES-CARRASQUILLO, Pedro A.; Douglas A. REYNOLDS; J.R. DELLER (2002):
"Language Identification using Gaussian Mixture Model Tokenization". In: *ICASSP 2002 Proceedings*, Vol. 1, May 2002.
(Zugriff: 29.05.2005, 16:47 MEZ)
<http://www.ll.mit.edu/IST/pubs/icassp02-ptorres.pdf>

WECHSLER, Martin; Paraic SHERIDAN; Peter SCHÄUBLE (1997):
"Multi-Language Text Indexing for Internet Retrieval". In: *Proceedings of the 5th RIAO Conference, Computer-Assisted Information Searching on the Internet*, Montreal, Canada.
(Zugriff: 28.09.2004, 16:19 MEZ)
<http://citeseer.ist.psu.edu/cache/papers/cs/6218/http:zSzzSzwww-ir.inf.ethz.chzSzDELOSzSzSheridanzSzsheridan-deloszSzriao97.pdf/multi-language-text-indexing.pdf >

WOMSER-HACKER, Christa (2003):
"Multilingualität im Information Retrieval". 3. Herbstschule "IR", Schloss Dagstuhl.

Internetquellen:

CEGLOWSKI, Maciej (2004):
Languid: a statistical language identifier.
(Ersch.: 2004. Zugriff 15.04.2005, 12:26 MEZ) <http://languid.cantbedone.org/>

EIDETICA: The Language Guesser. (Zugriff 14.04.2005, 17:44 MEZ)
<http://www.eidetica.com/services/guesser>

Global Internet Statistics (by Language).
(rev.: 30.09.2004. Zugriff 29.05.2005, 21:16 MEZ)
<http://www.glreach.com/globstats/index.php3>

HUFFMAN, Stephen (2000):
Unknown Language Identification. (Ersch.: 2000. Zugriff 15.04.2005, 17:10 MEZ)
<http://complingone.georgetown.edu/~langid/>

LANGUAGEIDENTIFIER (Download Seite). (Zugriff 14.04.2005, 16:55 MEZ)
<http://www.languageidentifier.com/>

LEXTEK Language Identifier SDK. (Ersch.: Zugriff 14.04.2005. 16:55 MEZ)
<http://www.lextek.com/langid/>

LINGUA::IDENT. (Zugriff 15.04.2005, 15:30 MEZ)
<http://search.cpan.org/~mpiotr/Lingua-Ident/Ident.pm>

MGUESSER (Download Seite). (Zugriff 15.04.2005, 16:00 MEZ)
<http://www.mnogosearch.org/guesser/>

MULINEX. Multilinguale Indexierungs- , Navigations- und Editier-Extensionen für das WWW. (rev.: 21.12.98. Zugriff 11.09.2005, 10:28 MEZ)
<http://mulinex.dfki.de/index-d.html>

PETAMEM: Sprachidentifikation. (rev.: 13.04.2005. Zugriff 14.04.2005, 17:15 MEZ)
<http://nlp.petamem.com/langident.cgi>

ROSETTE Language Identifier. (Ersch.: 2005. Zugriff 14.04.2005, 16:48 MEZ)
<http://www.basistech.com/products/text-processing/euclid.html>

SILC: Système d'Identification de la Langue et du Codage.
(Zugriff 14.04.2005, 16:10 MEZ)
<http://rali.iro.umontreal.ca>

TALENKNOBBEL. (Zugriff 14.04.2005, 17:36 MEZ)
<http://www.fuzzums.nl/talenknobbel/>

TEXTCAT Language Guesser Demo.
(rev.: 5.01.99. Zugriff 14.04.2005, 15:38 MEZ)
<http://odur.let.rug.nl/~vannoord/TextCat/Demo/>
XEROX: Language Identifier. (Zugriff 14.04.2005, 16:21 MEZ)
<http://www.xrce.xerox.com/competencies/content-analysis/tools/guesser>

http://de.wikipedia.org/wiki/Vektorraum_Retrieval (rev.: 25.04.2005. Zugriff 7.06.2005, 9:42 MEZ)

http://de.wikipedia.org/wiki/Skalarprodukt (rev.: 16.06.2005. Zugriff 23.06.2005, 17:56 MEZ)

http://de.wikipedia.org/wiki/Euklidischer_Raum (rev.: 24.05.2005. Zugriff 23.06.2005, 17:58 MEZ)

http://www.matheboard.de/lexikon/Markow-Prozess,definition.htm
(Zugriff 28.03.2005, 15:14 MEZ)

http://global-reach.biz/globstats/evol.html (rev.: 30.03.2005. Zugriff 6.07.2004, 17:51 MEZ)

http://de.wikipedia.org/wiki/Softwareentwicklungsprozess (rev.: 20.06.2005. Zugriff 3.07.2005, 21:57 MEZ)

Abbildungsverzeichnis

Tabellenverzeichnis

Inhalt der CD-ROM

Die beiliegende CD-ROM enthält:
1 – den Quellcode und die Dokumentation von LangIdent
2 – den authentischen multilingualen Testkorpus sowie die HTML-Dateien, aus denen er erstellt wurde
3 – den künstlich aufgebauten multilingualen Testkorpus
4 – den Trainingskorpus
5 – die beiden multilingualen Testkorpora

Anhang

Tabelle 1:
Fehlerquoten bei der Identifikation von sechs Sprachen differenziert nach der Sprache, der Klassifikationsmethode und der Dokumentgröße (erster Testkorpus, erste Version der wortbasierten Methode)

Sprache	Dokument-Größe	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
Deutsch	50 Byte	*9,24%*	*6,02%*	*1,97%*	*23,14%*
	100 Byte	*2,24%*	*1,17%*	*0,34%*	*6,52%*
	250 Byte	*0 %*	*0%*	*0%*	*0,35%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Englisch	50 Byte	*3,18%*	*5,36%*	*1,85%*	*10,95%*
	100 Byte	*0,15%*	*0,87%*	*0,15%*	*1,75%*
	250 Byte	*0,%*	*0,12%*	*0,12%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Spanisch	50 Byte	*5,87%*	*10,8%*	*5,9%*	*16,14%*
	100 Byte	*1,25%*	*4%*	*1%*	*2,65%*
	250 Byte	*0%*	*0,12%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Italienisch	50 Byte	*17,54%*	*10,45%*	*2,28%*	*28,91%*
	100 Byte	*5,71%*	*3,01%*	*0,25%*	*15,39%*
	250 Byte	*0,73%*	*0,12%*	*0%*	*3,55%*
	500 Byte	*0%*	*0%*	*0%*	*0,72%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Französisch	50 Byte	*27,32%*	*11,1%*	*4,7%*	*44,32%*
	100 Byte	*11,56%*	*1,93%*	*0,81%*	*33,76%*
	250 Byte	*2,09%*	*0,12%*	*0%*	*21,89%*
	500 Byte	*0,24%*	*0%*	*0%*	*18,49%*
	1000 Byte	*0%*	*0%*	*0%*	*10,14%*
Russisch	50 Byte	*0%*	*0%*	*0%*	*25,44%*
	100 Byte	*0%*	*0%*	*0%*	*7,57%*
	250 Byte	*0%*	*0%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Durchschnitt	50 Byte	*10,52%*	*7,29%*	*2,78%*	*24,82%*
	100 Byte	*3,48%*	*1,83%*	*0,42%*	*11,27%*
	250 Byte	*0,47%*	*0.08%*	*0,02%*	*4,36%*
	500 Byte	*0.04%*	*0%*	*0%*	*3,2%*
	1000 Byte	*0%*	*0%*	*0%*	*1,69%*

Tabelle 2:
Fehlerquoten bei der Identifikation von sechs Sprachen differenziert nach der Sprache, der Klassifikationsmethode und der Dokumentgröße (erster Testkorpus, modifizierte wortbasierte Methode)

Sprache	Dokument-Größe	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
Deutsch	50 Byte	*9,24%*	*6,02%*	*1,97%*	*21,59%*
	100 Byte	*2,24%*	*1,17%*	*0,34%*	*4,63%*
	250 Byte	*0 %*	*0%*	*0%*	*0%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Englisch	50 Byte	*3,18%*	*5,36%*	*1,85%*	*11,01%*
	100 Byte	*0,15%*	*0,87%*	*0,15%*	*1,7%*
	250 Byte	*0,%*	*0,12%*	*0,12%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Spanisch	50 Byte	*5,87%*	*10,8%*	*5,9%*	*15,4%*
	100 Byte	*1,25%*	*4%*	*1%*	*3,1%*
	250 Byte	*0%*	*0,12%*	*0%*	*0,48%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Italienisch	50 Byte	*17,54%*	*10,45%*	*2,28%*	*22,99%*
	100 Byte	*5,71%*	*3,01%*	*0,25%*	*5,71%*
	250 Byte	*0,73%*	*0,12%*	*0%*	*0%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Französisch	50 Byte	*27,32%*	*11,1%*	*4,7%*	*28,9%*
	100 Byte	*11,56%*	*1,93%*	*0,81%*	*8,31%*
	250 Byte	*2,09%*	*0,12%*	*0%*	*0,62%*
	500 Byte	*0,24%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Russisch	50 Byte	*0%*	*0%*	*0%*	*25,44%*
	100 Byte	*0%*	*0%*	*0%*	*7,57%*
	250 Byte	*0%*	*0%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Durchschnitt	50 Byte	*10,52%*	*7,29%*	*2,78%*	*20,89%*
	100 Byte	*3,48%*	*1,83%*	*0,42%*	*5,17%*
	250 Byte	*0,47%*	*0.08%*	*0,02%*	*0,22%*
	500 Byte	*0.04%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*

Tabelle 3:
Fehlerquoten bei der Identifikation von sieben Sprachen (nach dem Hinzufügen von Tschechisch) differenziert nach der Sprache, der Klassifikationsmethode und der Dokumentgröße (erster Testkorpus)

Sprache	Dokument-Größe	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
Deutsch	50 Byte	*9,31%*	*6,02%*	*1,97%*	*21,61%*
	100 Byte	*2%*	*1,17%*	*0,34%*	*4,63%*
	250 Byte	*0 %*	*0%*	*0%*	*0%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Englisch	50 Byte	*4,13%*	*5,39%*	*1,85%*	*13,37%*
	100 Byte	*0,48%*	*0,87%*	*0,15%*	*2,71%*
	250 Byte	*0,%*	*0,12%*	*0,12%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Spanisch	50 Byte	*7,24%*	*10,87%*	*5,9%*	*16,48%*
	100 Byte	*1,45%*	*4%*	*1%*	*3,5%*
	250 Byte	*0%*	*0,12%*	*0%*	*0,48%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Italienisch	50 Byte	*19,54%*	*10,48%*	*2,31%*	*24,78%*
	100 Byte	*6,73%*	*3,01%*	*0,25%*	*6,57%*
	250 Byte	*0,86%*	*0,12%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Französisch	50 Byte	*26,92%*	*11,18%*	*4,72%*	*29,14%*
	100 Byte	*11,4%*	*1,93%*	*0,81%*	*8,36%*
	250 Byte	*2,34%*	*0,12%*	*0%*	*0,62%*
	500 Byte	*0,49%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Russisch	50 Byte	*0%*	*0%*	*0%*	*25,44%*
	100 Byte	*0%*	*0%*	*0%*	*7,57%*
	250 Byte	*0%*	*0%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Tschechisch	50 Byte	*4,33%*	*4,54%*	*5,56%*	*28,48%*
	100 Byte	*1,82%*	*1,01%*	*1,67%*	*9,78%*
	250 Byte	*0%*	*0,12%*	*0,25%*	*0,62%*
	500 Byte	*0%*	*0%*	*0%*	*0,24%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Durchschnitt	50 Byte	*10,21%*	*6,92%*	*3,19%*	*22,76%*
	100 Byte	*3,41%*	*1,71%*	*0,6%*	*6,16%*
	250 Byte	*0,46%*	*0,08%*	*0,05%*	*0,3%*
	500 Byte	*0,07%*	*0%*	*0%*	*0,03%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*

Tabelle 4:
Fehlerquoten bei der Identifikation von acht Sprachen (nach dem Hinzufügen von Ukrainisch) differenziert nach der Sprache, der Klassifikationsmethode und der Dokumentgröße (erster Testkorpus)

Sprache	Dokument-Größe	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
Deutsch	50 Byte	*9,39%*	*6,02%*	*1,97%*	*21,61%*
	100 Byte	*2%*	*1,17%*	*0,34%*	*4,63%*
	250 Byte	*0 %*	*0%*	*0%*	*0%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Englisch	50 Byte	*4,16%*	*5,39%*	*1,85%*	*13,37%*
	100 Byte	*0,48%*	*0,87%*	*0,15%*	*2,71%*
	250 Byte	*0,%*	*0,12%*	*0,12%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Spanisch	50 Byte	*7,03%*	*10,87%*	*5,9%*	*16,48%*
	100 Byte	*1,45%*	*4%*	*1%*	*3,5%*
	250 Byte	*0%*	*0,12%*	*0%*	*0,48%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Italienisch	50 Byte	*19,68%*	*10,48%*	*2,31%*	*24,78%*
	100 Byte	*6,88%*	*3,01%*	*0,25%*	*6,57%*
	250 Byte	*0,98%*	*0,12%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Französisch	50 Byte	*27,08%*	*11,18%*	*4,72%*	*29,14%*
	100 Byte	*11,56%*	*1,93%*	*0,81%*	*8,36%*
	250 Byte	*2,71%*	*0,12%*	*0%*	*0,62%*
	500 Byte	*0,49%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Russisch	50 Byte	*4,73%*	*2,47%*	*1,66%*	*30,58%*
	100 Byte	*1,59%*	*0,37%*	*0,16%*	*11,17%*
	250 Byte	*0,12%*	*0,12%*	*0%*	*0,74%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Tschechisch	50 Byte	*4,33%*	*4,54%*	*5,56%*	*28,48%*
	100 Byte	*1,82%*	*1,01%*	*1,67%*	*9,78%*
	250 Byte	*0%*	*0,12%*	*0,25%*	*0,62%*
	500 Byte	*0%*	*0%*	*0%*	*0,24%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Ukrainisch	50 Byte	*8,5%*	*4,32%*	*4,54%*	*31,8%*
	100 Byte	*5,57%*	*0,72%*	*1,02%*	*11,95%*
	250 Byte	*2,47%*	*0%*	*0%*	*1,23%*
	500 Byte	*0,24%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
	50 Byte	*10,61%*	*6,91%*	*3,56%*	*24,53%*

Durchschnitt	100 Byte	*3,92%*	*1,63%*	*0,67%*	*7,33%*
	250 Byte	*0,78%*	*0,09%*	*0,05%*	*0,49%*
	500 Byte	*0,09%*	*0%*	*0%*	*0,03%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*

Tabelle 5:
Fehlerquoten bei der Identifikation von acht Sprachen differenziert nach der Sprache, der Klassifikationsmethode und der Dokumentgröße (zweiterTestkorpus)

Sprache	Dokument-Größe	Klassifikationsmethoden			
		Vector Space	Ad Hoc	Bayes'	Wortbasiert
Deutsch	50 Byte	*8,3%*	*6,64%*	*2,32%*	*20,04%*
	100 Byte	*1,9%*	*1,96%*	*0,21%*	*4,48%*
	250 Byte	*0,12 %*	*0,12%*	*0%*	*0%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Englisch	50 Byte	*4,5%*	*6,72%*	*1,79%*	*12,4%*
	100 Byte	*0,82%*	*1,01%*	*0,1%*	*2,26%*
	250 Byte	*0,%*	*0%*	*0 %*	*0%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Spanisch	50 Byte	*5,37%*	*10,39%*	*5,55%*	*15,76%*
	100 Byte	*1,25%*	*3,76%*	*0,85%*	*3,91%*
	250 Byte	*0%*	*0,37%*	*0%*	*0,37%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Italienisch	50 Byte	*20,26%*	*11,1%*	*1,94%*	*24,68%*
	100 Byte	*7,1%*	*2,27%*	*0,1%*	*6,6%*
	250 Byte	*0,48%*	*0,12%*	*0%*	*0,12%*
	500 Byte	*0%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Französisch	50 Byte	*29,45%*	*9,15%*	*3,53%*	*31,06%*
	100 Byte	*14,88%*	*2,02%*	*0,62%*	*10,32%*
	250 Byte	*3,14%*	*0,12%*	*0%*	*0,7%*
	500 Byte	*0,92%*	*0%*	*0%*	*0%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Russisch	50 Byte	*5,77%*	*2,84%*	*2,16%*	*31,18%*
	100 Byte	*2,26%*	*0,26%*	*0,16%*	*12,55%*
	250 Byte	*0,61%*	*0 %*	*0%*	*1,47%*
	500 Byte	*0%*	*0%*	*0%*	*0,24%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Tschechisch	50 Byte	*4,07%*	*3,51%*	*4,02%*	*20,98%*
	100 Byte	*1,28%*	*0,98%*	*1,18%*	*5,65%*
	250 Byte	*0,62%*	*0,25%*	*0,25%*	*0%*
	500 Byte	*0%*	*0%*	*0%*	*0 %*

	1000 Byte	*0%*	*0%*	*0%*	*0%*
Ukrainisch	50 Byte	*9,92%*	*6%*	*6,11%*	*31,32%*
	100 Byte	*6,46%*	*1,95%*	*2,2%*	*12,81%*
	250 Byte	*2,35%*	*0%*	*0,12%*	*1,36%*
	500 Byte	*1,22%*	*0 %*	*0,24%*	*0 %*
	1000 Byte	*0%*	*0%*	*0%*	*0%*
Durchschnitt	50 Byte	*10,95%*	*7,04%*	*3,43%*	*23,43%*
	100 Byte	*4,49%*	*1,78%*	*0,68%*	*7,32%*
	250 Byte	*0,91%*	*0,12%*	*0,05%*	*0,5%*
	500 Byte	*0,27%*	*0 %*	*0,03%*	*0,03%*
	1000 Byte	*0%*	*0%*	*0%*	*0%*

Eidesstattliche Erklärung

Eigenständigkeitserklärung nach §31 Abs. 5 RaPo

Hiermit erklären wir, dass wir die vorliegende Arbeit selbständig abgefasst und nicht anderweitig zu Prüfungszwecken verwendet haben. Weiterhin erklären wir, dass wir die Arbeit ausschließlich unter Verwendung der angegebenen Quellen und Hilfsmittel erstellt und alle wörtlichen und sinngemäßen Zitate aus diesen Quellen geeignet gekennzeichnet haben.

Hildesheim, den 22. August 2005